INTERNATIONAL DOCTORAL
SCHOOL OF THE USC

Congyu
Zhang Sprenger

PhD Thesis

PROCESS ORIENTED ONTOLOGY BASED ROBOTIC TASK PLANNING AND MODELLING

Santiago de Compostela, 2024

Doctoral Programme in Information Technology Research

ESCOLA DE DOUTORAMENTO
INTERNACIONAL DA USC

DOCTORAL THESIS

PROCESS ORIENTED ONTOLOGY BASED ROBOTIC TASK PLANNING AND MODELLING

Author

Congyu Zhang Sprenger

Supervisor/s: Juan Antonio Corrales Ramón, Norman Urs Baier

Tutor: Francisco Fernández Rivera

PHD PROGRAMME IN NFORMATION TECHNOLOGY RESEARCH

SANTIAGO

INDEX

1 SUMMARY IN GALICIAN 7

1.1 PLANIFICACIÓN E MODELIZACIÓN DE TAREFAS ROBÓTICAS BASEADAS EN ONTOLOXÍAS ORIENTADAS A PROCESOS 7

1.1.1 Motivación E Desenvol Vemento Da Investifación 7

1.1.2 Obxectivos De Investifación 10

1.1.3 Diseño Da Investigación 11

1.1.4 Metodoloxía 12

1.1.5 Publicacións 13

1.1.5.1 Publicación 1 13

1.1.5.2 Publicación 2 14

1.1.5.3 Publicación 3 14

1.1.5.4 Publicación 4 14

1.1.6 Conclusión 15

2 INTRODUCTION 16

2.1 MOTIVATION AND RESEARCH DEVELOPMENT 16

2.2 TECHNOLOGY BACKGROUND 19

2.2.1 Graphical Modelling/Programming Language for Robotic Processes 19

2.2.2 Ontology for Robotic Task Planning 20

2.2.3 Ontology Background 20

2.3 THESIS OUTLINE 21

3 RESEARCH OBJECTIVES 23

4 RESERCH METHODOLOGY 24

4.1 RESEARCH DESIGN 24

5 ROBOT PROCESS MODELLING AND NOTATION RTMN1.0 27

5.1 INTRODUCTION 27

5.2 LITERATURE REVIEW 27

5.2.1 Operators Facing Robotic Technology Challenges 27

5.2.1.1 Hard to Understand the Robots (error messages too technical, robotic processes too complex) 27

5.2.1.2 Reprogramming is Costly and Requires Expertise 28

5.2.1.3 Hard to Accept Robot Technology 29

5.2.2 Graphical Modelling/Programming Technologies 29

5.2.2.1 Visual Programming Languages (VPLs) 29

5.2.2.2 Behavior Tree Language (BTL) 30

5.2.3 BPMN Application in Robotics 30

5.3 PROPOSED METHOD 31

5.3.1 The Model 34

5.3.2 The Implementation 36

5.4 VALIDATION 38

5.5 CONCLUSION AND OUTLOOK 41

6 ROBOT PROCESS MODELLING AND NOTATION RTMN 2.0 FOCUS ON HUMAN ROBOT COLLABORATION 42

6.1 INTRODUCTION 42

6.2 LITERATURE REVIEW 42

6.2.1 What is Human–Robot Collaboration (HRC)? 42

6.2.2 The Importance of Human–Robot Collaboration (HRC) 43

6.2.3 Safety Standards and HRC Modes 44

6.2.3.1 Safety-rated Monitored Stop (SMS) 44

6.2.3.2 Hand-Guiding (HG) 45

6.2.3.3 Speed and Separation Monitoring (SSM) 45

6.2.3.4 Power and Force Limiting (PFL) 45

6.2.4 HRC Task Types 46

6.2.5 HRC Modelling Methods 46

6.2.5.1 Business Process Modelling Notation (BPMN) 46
6.2.5.2 Unified Modelling Language (UML) 47
6.2.5.3 Systems Modelling Language (SysML) 48
6.2.5.4 Behavior Trees 48
6.2.5.5 Petri Nets 49
6.2.6 Research Gap *49*
6.3 PROPOSED METHOD 50
6.3.1 The RTMN Elements *50*
6.3.2 RTMN 2.0 Elements *51*
6.3.3 RTMN 2.0 Sequence Flow Connection Rules *54*
6.3.4 The HRC Model *55*
6.3.4.1 Combining Collaboration Task Types and HRC Modes 55
6.3.4.1.1 Coexistence Fence (CF) 57
6.3.4.1.2 Sequential Cooperation SMS (SS) 57
6.3.4.1.3 Teaching HG (TH) 58
6.3.4.1.4 Parallel Cooperation SSM (PS) 58
6.3.4.1.5 Collaboration PFL (CP) 58
6.3.4.2 Workspace 60
6.3.4.3 Decision Making 60
6.3.5 Other Extensions and Modifications *62*
6.3.5.1 Requirements and KPI 62
6.3.5.2 Traceability of Requirements and KPIs 64
6.3.5.3 Robotic Process 64
6.3.5.4 Skill and Primitive 65
6.3.6 The Implementation *66*
6.3.7 Demonstration *66*
6.3.8 Validation *67*
6.3.9 Discussion *68*
6.3.10 Conclusions and Outlook *68*

7 ONTOLOGY SUPPORT ROBOT PROCESS MODELLING 69

7.1 INTRODUCTION 69
7.2 LITERATURE REVIEW AND PROBLEM DESCRIPTION 70
7.2.1 Literature Review *70*
7.2.1.1 Human-robot Collaboration 70
7.2.1.2 Autonomous Robotics 71
7.2.1.3 Industrial Manufacturing 71
7.2.2 Problem Definition *72*
7.3 METHODS - ONTOLOGY DEVELOPMENT 73
7.3.1 Ontology Purpose Identification and Requirements Elicitation (Phase1) *74*
7.3.1.1 Competency Questions 74
7.3.1.2 Modularization 75
7.3.2 Ontology Conceptualization and Formalization (Phase2&3) *76*
7.3.2.1 Conceptualization 76
7.3.2.1.1 Find the relevant terms for the ontology. 77
7.3.2.1.2 Define classes and their properties. 79
7.3.2.2 Formalization 81
7.3.2.2.1 SUMO Concepts 82
7.3.2.2.2 CORA(X) Concepts 83
7.3.2.2.3 ORPP Core Concepts and Axioms 84
7.3.2.2.4 PDDL Concepts 87
7.3.2.2.5 ACROBA Concepts 88
7.3.2.2.6 RTMN Concepts 88
7.4 ONTOLOGY IMPLEMENTATION (PHASE 4) 89
7.5 ONTOLOGY TESTING (PHASE 5) - THE USE OF ORPP IN THE ACROBA PROJECT 90
7.5.1 Populating the GUI (Orange Arrow 1.) *91*
7.5.2 Supporting task execution (Orange Arrow 2.) *91*
7.5.3 Supporting PDDL task planning (Orange Arrow 3.) *91*
7.5.5 PDDL Task Planning Application- Example of Orange Arrow 3 ..... *94*
7.5.5.1 Cell Setup 94
7.5.5.2 The Scenario 95

7.5.5.3 The PDDL (Planning Domain Definition Language) Definitions 96
7.5.5.4 Ontology Representation for PDDL Domain and Problem 98
7.6 CONCLUSION 99

8 FLEXIBLE TASK PLANNER USING RTMN2.0 AND ORPP ONTOLOGY 100

8.1 INTRODUCTION 100
8.2 LITERATURE REVIEW 100
8.2.1 ROS 100
8.2.2 Existing Task Planning Frameworks based on ROS 101
8.2.2.1 ROSPlan [161] 101
8.2.2.2 Plansys2 [162] 101
8.2.2.3 SkiROS2 [26] 101
8.2.2.4 Navigation 2 [164] 101
8.2.3 Behavior Tree 102
8.2.3.1 Navigation2 102
8.2.3.2 Groot 103
8.3 PROPOSED METHOD 103
8.3.1 The Task Planner 104
8.3.1.1 ROS2 and the Bridge with ROS1 104
8.3.1.2 Behavior Tree Writer 105
8.3.1.3 PDDL [153] 105
8.3.2 The Interfaces 105
8.3.2.1 Skills 106
8.3.2.2 HMI 107
8.3.2.3 GUI 107
8.3.2.4 Virtual Gym and DRL Modules 108
8.3.2.5 Ontology 108
8.4 FUTURE ADD-ONS 109
8.4.1 Assembly Sequence Planner 109
8.4.2 Direct Access to Primitives for Task Duration Optimization 109
8.5 CONCLUSION 109

9 RESULTS - EXPERIMENTAL VALIDATIONS 110

9.1 INTRODUCTION 110
9.2 SCENARIO 1: ROBOT PROGRAMMING THROUGH THE GUI USING RTMN 110
9.2.1 Process/Task Description 111
9.2.2 Modelling in the GUI 112
9.2.3 Robot Demonstration 117
9.3 SCENARIO 2: TASK PLANNING USING PDDL AND THE ORPP ONTOLOGY 121
9.3.1 The Setup 121
9.3.1.1 The Initial State 122
9.3.1.2 The Goal State 122
9.3.1.3 The Constraints 123
9.3.2 PDDL (Planning Domain Definition Language) Definitions 123
9.3.2.2 PDDL Problem Description 123
9.3.3 Ontology Supporting PDDL 124
9.3.3.1 Ontology represented in OWL 124
9.3.3.2 SPARQL Queries 125
9.4 SCENARIO 3: HUMAN ROBOT COLLABORATION USING RTMN 2.0 127
9.4.1 The HRC Task/Process 127
9.4.2 Speed and Separation Monitoring 127
9.4.3 Human Speed Monitoring 129

10 CONCLUSION 130

11 BIBLIOGRAPHY 132

12 ACKNOWLEDGMENT 140

13 INDEX OF FIGURES 141

14 INDEX OF TABLES 143

15 APPENDICES 143

15.1 ACROBA USE CASES 143
15.2 USER QUESTIONNAIRE 146
15.3 ANSWERS OF THE USER QUESTIONNAIRE 150
15.4 ANSWERS FROM USER INTERVIEW 156
15.5 ANSWERS FROM PROTOTYPE VALIDATION 158
15.6 VALIDATION FORM FOR GUI VERSION 2.0 159
15.7 ONTOLOGY FOR PDDL DOMAIN AND PROBLEM DESCRIPTION 159
15.8 QUESTIONNAIRE RESULTS 161
15.9 PUBLICATIONS AND AUTHOR CONTRIBUTIONS 163
15.9.1 Publication 1 163
15.9.2 Publication 2 164
15.9.3 Publication 3 165
15.9.4 Publication 4 167

LIST OF ABBREVIATIONS

Abbreviation	Definition
API	Application Programming Interface
ASP	Assembly Sequence Planner
BPMN	Business Process Model and Notation
BT	Behavior Tree
BTL	Behavior Tree Language
CAD	Computer Aided Design
CF	Coexistence Fence
CORA	Core Ontology for Robotics and Automation
CP	Collaboration PFL
CPN	Colored Petri Nets
CQ	Competency Question
CSF	Critical Success Factor
DAWG	Data Access Working Group
DL	Description Logic
DOI	Digital Object Identifier
DOLCE	Descriptive Ontology for Linguistic and Cognitive Engineering
DRL	Deep Reinforcement Learning
DUL	DOLCE+DnS Ultralite
EU	European Union
FOL	First Oder Logic
GUI	Graphical User Interface
HG	Hand-guiding
HMI	Human Machine Interface
HRC	Human Robot Collaboration
HRI	Human Robot Interaction
AI	Artificial intelligence
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
KB	Knowledge Base
KPI	Key performance indicator
MATE	Measure, Adapt and Teach
MPMN	Manufacturing Process Modelling Notation
MPMS	Manufacturing Process Management System
MRS	Multi-Robot System
NPC	non-player characters
OCRA	Ontologies for Human– Robot Collaboration
ORA	Ontology for Robotics and Automation
ORO	Autonomous Robot Architecture Ontology
ORPP	Ontology for Robotic Process Planning

OWL	Web Ontology Language
PCB	Printed Circuit Board
PDDL	Planning Domain Definition Language
PFL	Power and Force Limiting
PS	Parallel Cooperation SSM
PTH	Pin through Hole
RDF	Resource Description Framework
ROA	Robot Architecture Ontology
ROS	Robot Operating System
RTMN	Robot Task Modelling Notation
RTPO	Robot Task Planning Ontology
SME	Small and Medium-sized Enterprises
SMS	Safety-rated Monitored Stop
SOHO	Sharework Ontology for Human–Robot Collaboration
SPARQL	SPARQL Protocol and RDF Query Language
SS	Sequential Cooperation SMS
SSM	Speed and Separation Monitoring
SUMO	Suggested Upper Merged Ontology
SysML	Systems Modelling Language
TCUID	Task-Centered User Interface Design
TFD	Temporal Fast Downward
TH	Teaching HG
TRACE	Traceable Robotic Activity Composer and Executive
UML	Unified Modelling Language
VPL	Visual Programming Languages
WRC	Wireless Robotic Component
XML	Extensible Markup Language

1 SUMMARY IN GALICIAN

1.1 PLANIFICACIÓN E MODELIZACIÓN DE TAREFAS ROBÓTICAS BASEADAS EN ONTOLOXÍAS ORIENTADAS A PROCESOS

1.1.1 Motivación E Desenvolvemento Da Investigación

A miña área de investigación de doutoramento orixínase no proxecto EU ACROBA HORIZON 2020 [1]. ACROBA é un proxecto financiado pola UE no que participan 17 entidades de 9 países. "O proxecto ACROBA ten como obxectivo desenvolver e demostrar un novo concepto de plataformas de robótica cognitiva baseado nun enfoque modular capaz de adaptarse facilmente a practicamente calquera escenario industrial aplicando principios de fabricación áxil. " [2].

En ACROBA identifícase o seguinte reto:

> *O aumento dos custos laborais ou mesmo a escaseza de man de obra cualificada amplía a área na que se poden empregar os robots de forma rendible. Non obstante, para superar a barreira psicolóxica que impide a adopción efectiva da robótica, os robots teñen que estar programados de forma que os traballadores habituais dunha peme común poidan dominalos. [2]*

A miña motivación para a investigación é resolver este desafío e permitir que expertos non programadores controlen robots dun xeito sinxelo.

Aínda que na proposta do proxecto ACROBA se identificaron problemas xerais da industria, para dar resposta ás necesidades de programación en termos de flexibilidade, facilidade de uso, intelixencia e requisitos de código aberto, foi necesario realizar unha revisión bibliográfica máis completa e sistemática.

Para establecer unha brecha de investigación precisa, comecei a miña revisión da literatura. Plantei tres hipóteses e formulei tres preguntas de investigación:

- Hipótese 1

Existe unha brecha entre a complexa programación de robots e os usuarios que non teñen coñecementos de programación.

- Hipótese 2

Necesítase unha interface de planificación e control do robot fácil de entender, con representacións intuitivas das tarefas robóticas, que proporcione aos interesados humanos unha comprensión natural dos procesos robóticos.

- Hipótese 3

Un modelo e unha notación tipo BPMN orientados a procesos centrados no dominio da robótica crearán unha ponte estandarizada para a brecha entre o deseño e a implementación do proceso robótico.

- Pregunta de investigación 1

Cales son os principais problemas aos que se enfrontan os operadores ao utilizar os actuais sistemas robóticos na fabricación áxil?

- Pregunta de investigación 2

Cales son as tecnoloxías de modelado/programación gráfica utilizadas para planificar e executar tarefas robóticas?

- Pregunta de investigación 3

Cales son as aplicacións actuais dos modelos e notacións orientadas a procesos no campo da robótica?

Tras estas preguntas de investigación, realicei varias quendas de revisións bibliográficas (as revisións bibliográficas detalladas preséntanse nos capítulos 2, 3, 4 e 5). Na literatura, atopei que é difícil para os operadores entender o software robótico (mensaxes de erro demasiado técnicas, procesos robóticos demasiado complexos). A reprogramación é moi cara e require unha experiencia importante. A literatura tamén revelou que, aínda que existen marcos e ferramentas dispoñibles para simplificar a programación do robot, estas ferramentas adoitan ser demasiado técnicas e non son o suficientemente intuitivas para os operadores e enxeñeiros. Adoitan ser específicos de marca, sen posibilidade de cambiar dun provedor a outro. Ademais, mesmo cando o sistema robot está programado e listo, o seu uso segue esixindo moitos coñecementos. Noutras palabras, os expertos non en robótica afrontan desafíos na industria de fabricación áxil de rápido crecemento. Por unha banda, a personalización masiva require unha programación flexible de robots, que é lento, custosa e require altos niveis de especialización. Por outra banda, os sistemas actuais son difíciles de comprender e controlar. Aínda que os sistemas robóticos industriais modernos son cada vez máis intelixentes e flexibles, normalmente están deseñados para aplicacións específicas a gran escala, o que fai que a súa implementación sexa demasiado complexa e custosa para as pemes [3]. A figura 1 ilustra a brecha de investigación atopada na literatura.

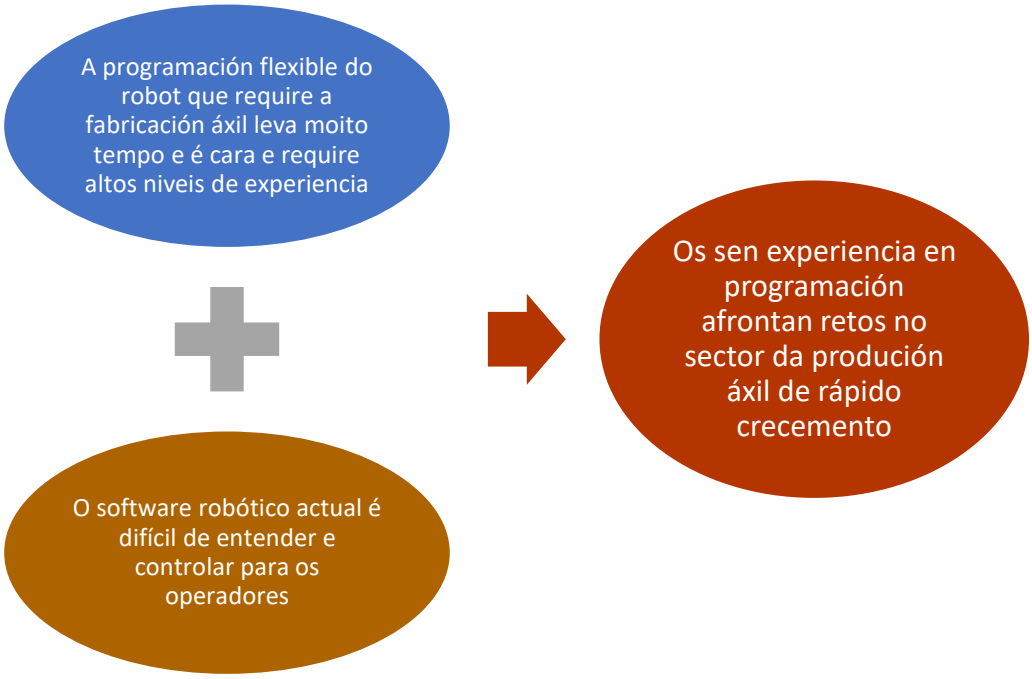


Figura 1. Conclusións da revisión da literatura

Para propoñer unha solución tecnolóxica axeitada para salvar esta brecha, realicei a miña investigación de métodos mixtos nas cinco liñas piloto industriais a gran escala de ACROBA.

Existen cinco grandes liñas piloto no proxecto ACROBA para a súa implementación e demostración (ver anexo 12.1). Considéranse representativos da produción áxil porque os seus

procesos son liñas de produción a gran escala que pasan da produción en masa á personalización masiva co obxectivo de proporcionar unha resposta rápida ao cliente, transformando a velocidade e a axilidade nunha vantaxe competitiva fundamental. O obxectivo destes casos de uso é utilizar robots para automatizar parcial ou totalmente os seus procesos de produción, sendo ACROBA o facilitador. Para analizar os requisitos e necesidades dos teus procesos, deseña e enviei cuestionarios para recoller os requisitos dos usuarios (os cuestionarios e as respostas preséntanse nos Anexos 12.2 e 12.3 da tese). Despois de analizar os resultados dos cuestionarios, realicei entrevistas a futuros usuarios de ACROBA destas cinco empresas para seguir analizando os requisitos. A análise final dos cuestionarios e das entrevistas revelou que os usuarios atoparon os seguintes problemas ao usar robots:

- Consideran que non teñen coñecementos especializados no uso de robots.
- Consideran que o sistema de control do robot é difícil de usar e comprender.
- Cando se produce un erro, non teñen suficiente información do sistema para axudarlles a resolver os problemas.

Os usuarios esperan ter un sistema de robot que:

- Sexa fácil de usar.
- Proporcionar unha interface gráfica de usuario con funcións sinxelas.
- Axúdaos en caso de erro.
- pode compensar a súa falta de coñecementos en programación e robótica.

A figura 2 mostra a brecha de investigación derivada do estudo da literatura, a investigación cuantitativa (cuestionarios) e cualitativa (entrevistas).

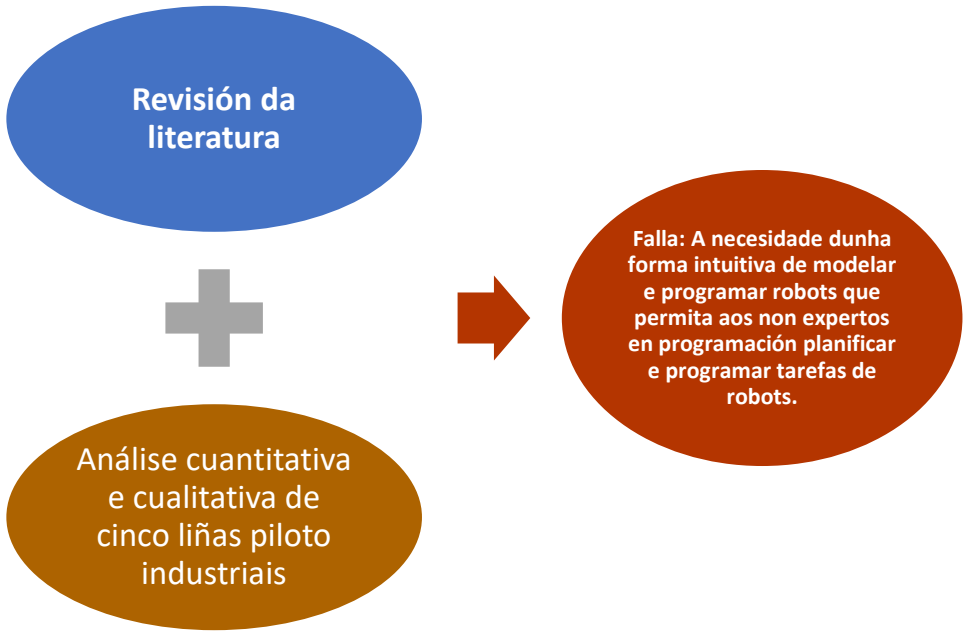


Figura 2. Fase de investigación

Todas as conclusións indican a necesidade dun modelado intuitivo e programación de tarefas de robótica, o que permite que os expertos non programadores poidan planificar e programar procesos do robot dunha forma máis sinxela. Para conseguilo, non só se necesita unha interface de usuario innovadora para a modelización, senón tamén un enfoque baseado no coñecemento para a programación de robots: é necesario adoptar unha ontoloxía para organizar a información en pezas de coñecemento reutilizables, proporcionar unha estrutura clara, garantir a precisión dos datos, permiten compartir coñecementos e apoian o razoamento [4-9].

1.1.2 Obxectivos De Investigación

En base ao anterior, identifiquei áreas potenciais para a miña investigación de doutoramento:

- Conectar procesos comerciais de alto nivel ao control de robots de nivel inferior
- Crear unha formalización común para a planificación de tarefas robóticas, de xeito que definicións como tarefa, competencia, primitiva, etc., se xeneralicen a partir das distintas definicións existentes na literatura, co obxectivo de estandarizar dende o nivel de proceso ata o de control do robot.
- Desenvolvemento dunha linguaxe de modelaxe baseada en ontoloxías para a planificación e execución de tarefas robóticas. Esta planificación de tarefas baseada en ontoloxías pretende proporcionar unha ferramenta fácil de usar que permita ao usuario compoñer procesos robóticos de forma rápida e económica. A característica comprensible pola máquina da ontoloxía desenvolvida nesta investigación serve como base común para a comunicación entre humanos e máquinas, compartindo a conceptualización do dominio do robot.
- Desenvolver unha ontoloxía de dominio que permita a reutilización do coñecemento e que permita aos non programadores planificar e programar tarefas do robot dun xeito flexible e sinxelo.
- Creación dunha ontoloxía de aplicación que abrangue a fabricación sen luz e a fabricación colaborativa no contexto do proxecto H2020 ACROBA[1].

As principais características desta linguaxe de modelado son

- Proporcionar unha interface gráfica de usuario (GUI) intuitiva e fácil de usar para deseñar, simular e executar procesos robóticos.
- Define a planificación simbólica de alto nivel para tarefas robóticas e crea a ponte para a planificación de movementos de baixo nivel para o control do robot.
- Integración de procesos de colaboración humano-robot.
- Utilización do "modelo de aceptación da tecnoloxía" para o deseño, desenvolvemento e validación da tecnoloxía

As principais características desta ontoloxía son:

- Combina o enfoque centrado na ferramenta (descríbese unha tarefa de fabricación en función dos compoñentes de hardware e software dispoñibles) e o enfoque centrado no produto (describindo o produto e os pasos de produción asociados independentemente dos recursos de produción específicos) [10].
- Inclúe ontoloxía de nivel superior, ontoloxía de nivel de dominio e ontoloxía de nivel de aplicación.
- Contén tres tipos de coñecemento na ontoloxía do dominio: coñecemento de procesos, coñecemento de planificación de tarefas e coñecemento de HRC. Integra datos de diferentes fontes, como cámaras, sensores, coñecemento do dominio e información de entrada humana nunha única ontoloxía [2].
- Acada a estandarización, a modularidade, a reutilización, a extensibilidade, a posta en común do coñecemento e o razoamento do coñecemento.

1.1.3 Diseño Da Investigación

Inicialmente, observei os diferentes tipos de métodos de investigación. Son moitas as metodoloxías de investigación que se poden adoptar neste estudo de doutoramento. Despois de comparar a fondo as vantaxes e os inconvenientes, decidín empregar a investigación de métodos mixtos, combinando enfoques cuantitativos e cualitativos (a investigación cuantitativa usa enquisas e cuestionarios para recoller datos numéricos mediante preguntas estruturadas; a investigación cualitativa utiliza entrevistas que permiten debates abertos para recoller información en profundidade). A investigación de métodos mixtos combínase coa creación de prototipos e probas. Presenta varias vantaxes que levan a unha comprensión máis ampla dun problema de investigación [11-14].

Existen varios deseños de investigación de métodos mixtos: deseño paralelo converxente, deseño secuencial explicativo e deseño secuencial exploratorio. Para establecer un estudo en profundidade e máis completo, seleccionouse para a miña investigación o método de deseño secuencial exploratorio. A visión xeral preséntase na Figura 3.

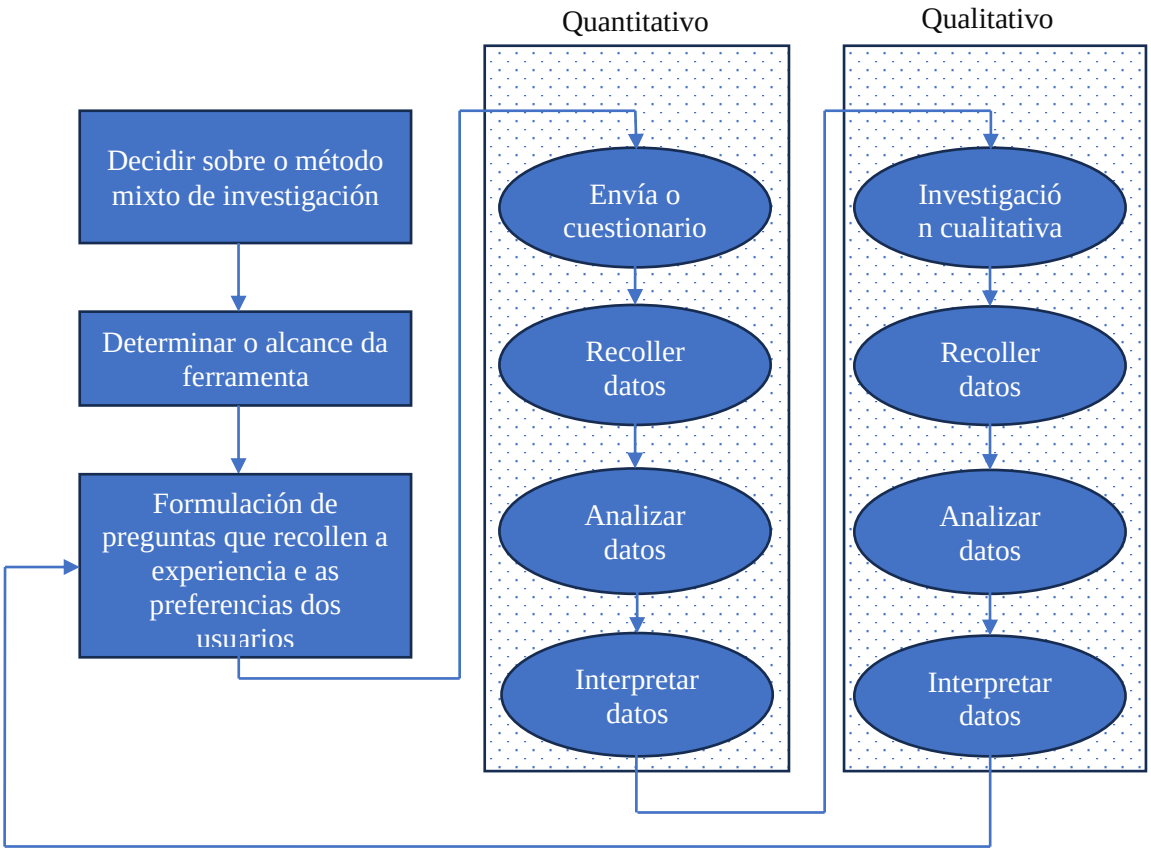


Figura 3. Deseño de investigación secuencial explicativa

A figura 3 mostra o meu proxecto de investigación. En primeiro lugar, determinei o obxectivo xeral da miña investigación de doutoramento. E despois, en función do obxectivo, formulei as miñas preguntas de investigación. Despois deste paso, escollín o método de investigación, que é un estudo de métodos mixtos. Co deseño de investigación de métodos mixtos, escollín o deseño de investigación secuencial explicativo. Primeiro realizouse a investigación cuantitativa para recoller datos cuantitativos. Despois da análise dos datos, realizouse a primeira rolda de conclusións/declaracións. Seguiuse a investigación cualitativa,

utilizando a entrevista como principal instrumento. Isto afondou nas conclusións/declaracións extraídas da primeira volta. Os resultados desta rolda tamén se aproveitaron para mellorar a formulación de preguntas que recollen a experiencia e as preferencias dos usuarios.

1.1.4 Metodoloxía

Establecín tres hipóteses e tres preguntas de investigación. Estes foron respondidos e tratados nas miñas catro publicacións (capítulo 2-5). A primeira publicación (capítulo 2) foi deseñada para confirmar a hipótese 1 (Hai unha brecha entre a programación complexa de robots e os usuarios que non teñen coñecementos de programación) e a hipótese 2 (Necesítase unha interface de planificación e control do robot fácil de entender, con representacións intuitivas). de tarefas robóticas, que proporciona unha comprensión natural dos procesos robóticos aos participantes humanos). Aínda que a RTMN abarca o modelado de centrada no modelado da colaboración entre robots e humanos como modelo xenérico (pregunta de investigación 1). Este feito foi confirmado polos datos recollidos a través de cuestionarios e entrevistas. Investíganse as tecnoloxías de modelado e programación existentes para responder ás preguntas de investigación 2 e 3. Estes mesmos modelos e tecnoloxías de programación proporcionaron a base para a linguaxe de modelado RTMN (Robot Task Modelling Notation), que foi deseñada para salvar a brecha entre o modelado de procesos e a execución do robot. para permitir aos usuarios sen coñecementos de programación planificar e controlar robots. O concepto clave de RTMN é modelar procesos robóticos cunha estrutura primitiva de tarefas-habilidades que divide os procesos robóticos en tarefas, habilidades do robot e primitivas de habilidades. Ademais, debido a que as notacións básicas de RTMN son similares ás que os usuarios usan para deseñar os seus fluxos de procesos de negocio, a familiaridade mellora o uso da ferramenta. A avaliación baseada no modelo de aceptación da tecnoloxía de Davis [15] indica un uso futuro positivo de RTMN (validación da hipótese 3: un modelo e notación orientado a procesos semellantes a BPMN, centrado no dominio da robótica, creará unha ponte estandarizada para a brecha entre o deseño e implementación do proceso robótico).

Aínda que a RTMN abarca o modelado de centrada no modelado da colaboración entre robots e humanos como modelo xenérico, non pode cubrir certos aspectos de seguridade. RTMN 2.0 (Capítulo 3) -a segunda publicación- é unha extensión e modificación de RTMN, enfocada a modelar a colaboración entre o home e o robot. Ademais, modelou requisitos e KPI para permitir a trazabilidade do proceso empresarial ata o nivel inferior de control robótico. As notacións relacionadas coa toma de decisións modifícanse como unha mellora da RTMN. A maior contribución de RTMN 2.0 é a creación de sinerxías entre os modos HRC (baseados nos estándares ISO) e os tipos de tarefas HRC existentes na literatura. Esta combinación deu lugar a cinco notacións de modelado HRC diferentes: Barreira de convivencia, Cooperación Secuencial SMS, Ensino HG, Cooperación Paralela SSM e Colaboración PFL. Para cada notación defínense criterios de colaboración e seguridade. Esta linguaxe de modelado está implantada nunha interface gráfica de usuario para ofrecer aos usuarios as seguintes vantaxes: maior intuitividade, modelos sinxelos de usar, sen código e garantía de seguridade e reutilización.

RTMN 2.0 facilita a programación de tarefas robóticas proporcionando unha interface intuitiva para modelar tarefas robóticas. Non obstante, para acadar maiores niveis de intelixencia na programación robótica, é necesario o apoio das tecnoloxías de IA. A ontoloxía xoga un papel importante na IA e na robótica, proporcionando datos estruturados, razoamento, comprensión de accións, conciencia do contexto, transferencia de coñecemento e aprendizaxe

semántica. O marco estruturado creado pola ontoloxía para a representación do coñecemento é fundamental para permitir un comportamento intelixente nos robots. Por estes motivos, ORPP (Ontology for Robotic Task Planning) foi desenvolvido como a miña terceira publicación (capítulo 4). ORPP aborda unha lagoa importante na literatura, que é a necesidade de cubrir a planificación de procesos robóticos de nivel superior e o control robótico de nivel inferior integrando as capacidades de representación e razoamento do coñecemento para optimizar a planificación de tarefas na fabricación áxil. Para que os sistemas baseados en ontoloxía sexan eficaces, a normalización do dominio é esencial. Esta ontoloxía abarca a representación do coñecemento para o dominio da planificación de tarefas robóticas e proporciona capacidades de razoamento.

RTMN 2.0 e a ontoloxía ORPP, como aplicacións separadas, non permiten a planificación holística de tarefas robóticas. Só poden alcanzar a súa plena capacidade cando están integrados para apoiar a planificación de tarefas. A última publicación (capítulo 5) presenta un enfoque completo para combinar a linguaxe de modelado RTMN 2.0 e a ontoloxía ORPP para a planificación de tarefas robóticas. RTMN 2.0 e ORPP xuntos pretenden apoiar e acelerar a planificación intuitiva e flexible de tarefas robóticas desde o front-end e o back-end. O planificador de tarefas proposto nesta publicación ten unha interface gráfica de usuario integrada que adopta os principios de modelado RTMN. En segundo plano, a ontoloxía ORPP serve como base de coñecemento e apoia a intelixencia do robot mediante consultas e razoamentos coa base de coñecemento. Este enfoque de combinar a planificación de tarefas, RTMN e ORPP maximiza as capacidades de planificación de tarefas para permitir a planificación de tarefas robótica da forma máis flexible posible.

A validación experimental (capítulo 6) presentada nesta tese mostra como funcionan conxuntamente as miñas contribucións de investigación e ofrece unha visión xeral das accións do robot. Preséntanse tres demostracións para validar a investigación que realicei no meu estudo de doutoramento. A primeira demostración utiliza a interface gráfica baseada na linguaxe de modelado RTMN para modelar un proceso robótico.A segunda demostración mostra como funciona a planificación de tarefas usando RTMN (GUI) e a ontoloxía ORPP. A terceira demostración ilustra un escenario de colaboración entre humanos e robots usando a notación de modelado desenvolvida en RTMN 2.0.

1.1.5 Publicacións

Publiquei catro artigos: 1) Planificador de tarefas flexible baseado en árbores de comportamento construído no marco ROS2; 2) Automatización de procesos robóticos con modelo e notación de tarefas de robot baseados en habilidades habilitados para ontoloxías (RTMN); 3) RTMN 2.0-Unha extensión de modelado e notación de tarefas de robots (RTMN) centrada na colaboración humano-robot; 4) ORPP - An Ontology for Skill-based Robotic Process Planning in Agile Manufacturing. Os detalles destas publicacións son os seguintes.

1.1.5.1 Publicación 1

Táboa 1. Detalles da publicación 1

Título	Robotic Process Automation with Ontology-enabled Skill-based Robot Task Model and Notation (RTMN)
Nome da conferencia/revista	IEEE RAAI 2022, Singapur, 09-11 de decembro de 2022
Autor	Congyu Zhang Sprenger, Thomas Ribeaud
Editora	IEEE
Ano de publicación	2022
Estado	Publicado
DOI	https://doi.org/10.1109/RAAI56146.2022.10092996

1.1.5.2 Publicación 2

Táboa 2. Detalles da publicación 2

Título	RTMN 2.0—An Extension of Robot Task Modelling and Notation (RTMN) Focused on Human–Robot Collaboration
Nome da conferencia/revista	Revista de Ciencias Aplicadas Número especial Tecnoloxías de IA para robots colaborativos e de servizo
Autor	Congyu Zhang Sprenger, Juan Antonio Corrales Ramón, Norman Urs Baier
Editora	MDPI
Ano de publicación	2023
Estado	Publicado
DOI	https://doi.org/10.3390/app14010283

1.1.5.3 Publicación 3

Táboa 3. Detalles da publicación 3

Título	ORPP – An Ontology for Skill-based Robotic Process Planning in Agile Manufacturing
Nome da conferencia/revista	Revista de electrónica Tema especial Aplicación da intelixencia artificial en robótica
Autor	Congyu Zhang Sprenger, Juan Antonio Corrales Ramón, Norman Urs Baier
Editora	MDPI
Ano de publicación	2024
Estado	Publicado
DOI	https://doi.org/10.3390/electronics13183666

1.1.5.4 Publicación 4

Táboa 4. Detalles da publicación 4

Título	Behavior Trees based Flexible Task Planner Built on ROS2 Framework
Nome da conferencia/revista	ETFA Stuttgart setembro de 2022
Autor	Congyu Zhang Sprenger, Thomas Ribeaud
Editora	IEEE
Ano de publicación	2022
Estado	Publicado
DOI	https://doi.org/10.1109/ETFA52439.2022.9921683

1.1.6 Conclusión

Esta tese describe o proceso do meu doutoramento e a contribución da investigación que realicei. O obxectivo principal da miña investigación é resolver os retos aos que se enfrontan os PEMEs á hora de implementar solucións robóticas na fabricación áxil. Os principais retos identificados son a adquisición de coñecementos especializados en robótica, necesarios para comprender a alta complexidade dos sistemas robóticos existentes, e a adquisición de coñecementos exhaustivos subxacentes aos sistemas robóticos, así como a redución dos elevados custos de programación para a súa implantación. As miñas principais contribucións consisten en paliar estes retos mediante a creación dunha linguaxe de modelado intuitiva para planificar e controlar procesos robóticos co apoio de ontoloxías. Isto permite aos expertos non robótica programar e reprogramar robots de forma flexible, así como adquirir coñecementos relevantes para lograr unha personalización masiva.

Conseguíronse os obxectivos de investigación deste estudo de doutoramento. Non obstante, pódense realizar máis investigacións para desenvolver características adicionais. RTMN é intuitivo e flexible para modelar procesos de complexidade simple a media. Non obstante, para procesos complexos que teñen, por exemplo, varios ciclos, a RTMN carece de funcionalidades/elementos de modelado que simplifiquen o modelado de procesos. Outras extensións deberían centrarse en engadir estes elementos ás notacións de modelado. ORPP estableceu unha base sólida para a ontoloxía de planificación de tarefas robóticas e pódese usar para estenderse a outras ontoloxías de dominio. Afondar a colaboración entre humanos e robots, aumentar o coñecemento relacionado coa seguridade e os factores humanos e ampliar a planificación do PDDL a escenarios xenéricos e máis complexos son áreas de investigación futura. Ademais, a conexión/alineamento co Sistema Operativo de Robots (ROS) fará que a ontoloxía ORPP sexa máis adaptable e poderosa (ROS é un conxunto de bibliotecas de software e ferramentas que axudan a crear aplicacións para robots [16]). Polo tanto, o desenvolvemento e a mellora futuras deberían centrarse nestas áreas mencionadas anteriormente.

2 INTRODUCTION

2.1 MOTIVATION AND RESEARCH DEVELOPMENT

The PhD research area of this thesis comes from the EU HORIZON 2020 ACROBA project [1]. ACROBA is an EU funded project that involves 17 entities from 9 countries. “The ACROBA project aims to develop and demonstrate a novel concept of cognitive robotic platforms based on a modular approach able to be smoothly adapted to virtually any industrial scenario applying agile manufacturing principles.” [2].

In ACROBA, the following challenge is identified:

> *Increasing labor costs or indeed a distinct skilled labor shortage broadens the zone in which robots can be deployed profitably. Yet to overcome the psychological barrier to effectively adopt robotics, robots need to be programmed in a way, such that regular employees of a common SME can master it. [2]*

The goal of this research is to resolve this challenge and allow non-programming experts to be able to control robots in an easy way.

Although the general problems in industry were identified in the proposal of the ACROBA project, to address programming needs for flexibility, user friendliness, intelligence and open-source requirements, a more thorough and systematic literature review had to still to be conducted.

To establish precise research gap, the thesis starts with a literature review. There are three hypotheses and formulated three research questions:

- Hypothesis 1

There is a gap between the complex robot programming and users lacking expertise in programming.

- Hypothesis 2

There is a need for an easy-to-understand robot planning and controlling interface with intuitive robotic task representations, which provides a natural understanding of robotic processes to human actors.

- Hypothesis 3

An intuitive low-code process-oriented model and notation that focuses on the robotic domain will create a standardized bridge for the gap between the robotic process design and implementation.

- Research Question 1

What are the major problems operators have when using the current robotic systems in agile manufacturing?

- Research Question 2

What are the technologies for graphical modelling/programming that are used for robotic task planning and execution?

- Research Question 3

What are the current applications of process-oriented model and notation in the robotics domain?

Following these research questions, the author conducted several rounds of literature reviews (the detailed literature reviews are presented in Chapter 5, 6, 7, 8). The literature

showed that it is hard for operators to understand robotic software (error messages too technical, robotic processes too complex). Reprogramming is very expensive and requires significant expertise. The literature also established that although there are frameworks and tools available to simplify robot programming, these tools are often too technical and not intuitive enough for operators and engineers. They are often brand specific, with no possibility of changing from one supplier to another. Furthermore, even when the robot system is programmed and all set, using the system still requires a lot of expertise. In other words, non-robotic experts are facing challenges in the fast-growing agile production industry. On the one hand mass-customization requires flexible robot programming which is time consuming, costly, and requires high levels of expertise. On the other hand, current systems are difficult to understand and controlled. Although modern industrial robotic systems become smarter and more flexible, they are normally tailored for specific, large-scale applications, which makes their implementation too complex and costly for SME [3]. Figure 4 illustrates the research gap found in literature.

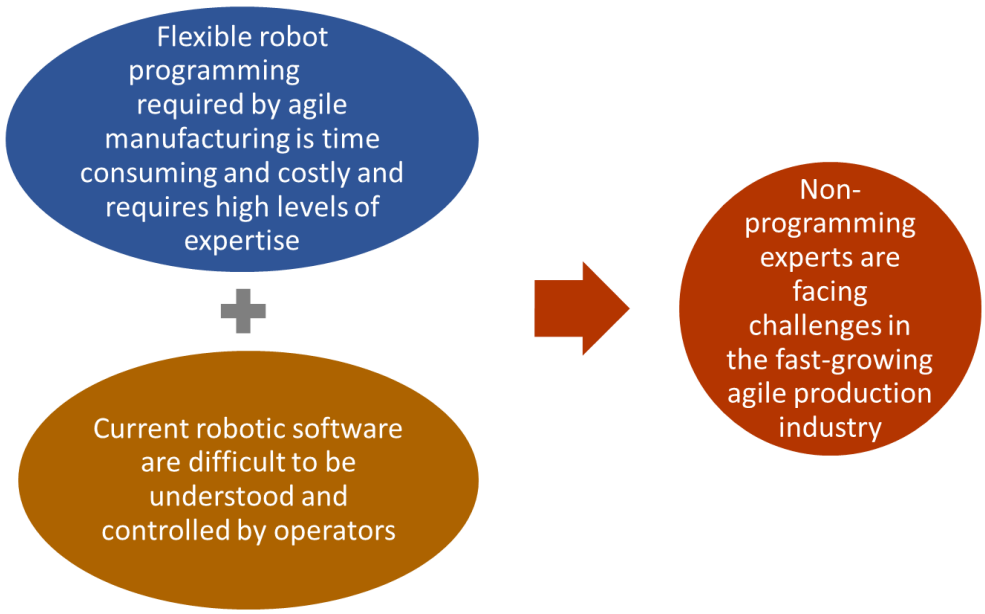


Figure 4. Literature Review Finding

In order to propose a suitable technological solution for filling this gap, a mix-methods approach on the five industrial large-scale pilot lines in ACROBA was used.

There are five large pilot lines in the ACROBA project for implementation and demonstration (see Appendix 15.1). They are considered representative of agile manufacturing because their processes are large-scale manufacturing lines that are transforming from mass-production to mass-customization with the aim of rapid response to the customer – turning speed and agility into a key competitive advantage. The goal of these use cases is to utilize robots to partially or fully automate their production processes with ACROBA as the enabler. To analyze their process requirements and needs, questionnaires were designed and sent out questionnaires to gather user requirements (the questionnaires and answers are presented in Appendix 15.2 and 15.3 of the thesis). After analyzing the questionnaire results, interviews with the future users of ACROBA from these five companies were conducted to deepen the

requirement analysis. The final analysis of the questionnaires and interviews showed the users encountered the following problems when using robots:

- They feel they are lacking expertise in using robots.
- They find the robot control system difficult to use and hard to understand.
- When there is an error, they have insufficient information from the system helping them to solve the problems.

The users expect to have a robot system that:

- Is user friendly and easy to use.
- Provides graphical user interface with easy functionalities.
- Can help them when error occurs.
- Can compensate for their lack of expertise in programming and robotics.

Figure 5 shows the research gap derived from literature study, quantitative (questionnaires) and qualitative (interviews) research.

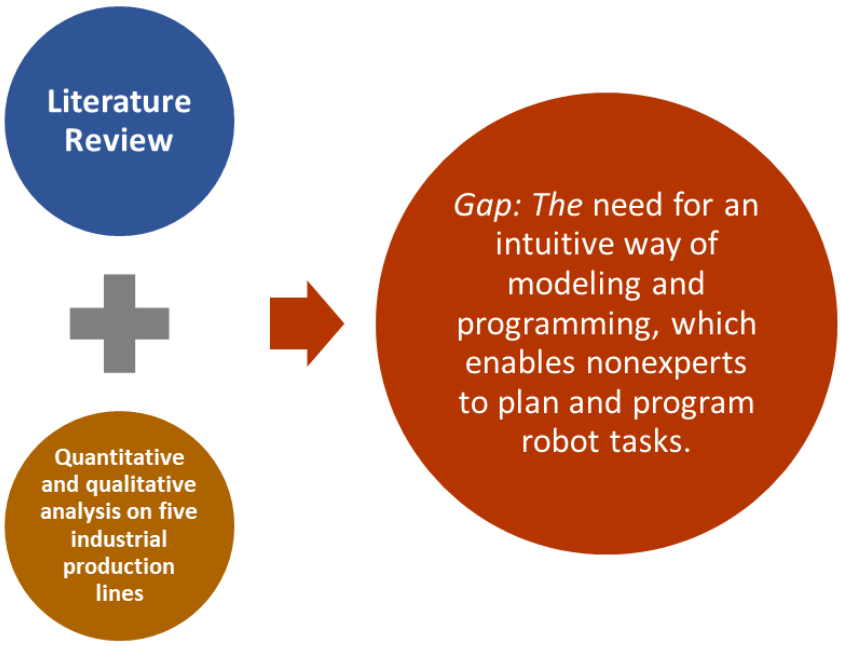


Figure 5. Research Gap

All the findings indicate the need for intuitive modelling and programming of robotics tasks, which enables non-programming experts to plan and program robot processes in an easier way. To do so, not only an innovative user interface for modelling is needed, but also a knowledge-enabled approach to robot programming – ontology needs to be adopted to organize information in re-usable pieces of knowledge, provide clear structure, ensure data accuracy, enable knowledge sharing and supports reasoning [4-9].

2.2 TECHNOLOGY BACKGROUND

2.2.1 Graphical Modelling/Programming Language for Robotic Processes

There are many frameworks facilitating the development of robotics systems, however, high programming skills are required [17]. A.G. Billard, S. Calinon, and R. Dillmann [18] pointed out that important research efforts had been deployed to simplify the programming of robots.

Currently, many Visual Programming Languages (VPLs) such as Blockly, Thymio, Intera Studio, Scratch, Kodu and Wonder, which provide a graphical computer programming interface aiming at making programming easier. These VPLs are used for various purposes, some for kids' games, some for robotic applications development, others for general-purpose programming [19]. Ahmad and Idrees pointed out that these VPL are good for teaching basic programming concepts, however, they lack fundamental programming capabilities [19]. There are other interfaces that are more user oriented. For example, C. Lewis and J. Rieman have proposed the Task-Centered User Interface Design (TCUID) approach [20], as Villani etc. developed the MATE (Measure, Adapt and TEach) framework [21].

Google's Blockly is an open-source framework provided with visual block-based icons and a drag-drop programming environment [22]. Although Blockly's puzzle-like and code reuse features are considered advantageous [23], it is still difficult for the non-programmers to understand the subroutines [24].

[25] pointed out that the Behavior Trees language has achieved huge success in the computer game AI community, and they are gaining popularity rapidly in robotics [26]. They believe that modularity is the main reason. It is well known that modularity is a key enabler when designing complex, maintainable and reusable systems [25]. They described that due to the modular and the adaptable representation of robotic tasks, BTs are also used to enable non-experts to do robot programming. However, it is very hard for non-programmers to cope with their complexity.

Rovida et al. developed extended behavior trees for quick definition of flexible robotic tasks. They define tasks as sequences of robot skills and can simplify robot task programming [27]. The robot can be reconfigured quickly by providing its services to the human operator as goals and fulfils the task goals with the available skill set [7].

Regarding graphical modelling languages for processes, the Business Process Modelling and Notation (BPMN) language is a standard maintained by the Object Management Group [28]. It models business processes graphically. BPMN notations are highly expressive and flexible which makes the modelling activity easier. There are numerous discussions on the use of Business Process Management for combining the different emerging technologies [28], specifically in robotic area, there are some initiatives on modelling robot activities [17] [29] [30] [31], however, automatically generating ROS (Robot Operating System) code from models remains challenging.

P. Bocciarelli, A. D'Ambrogio, A. Giglio, and E. Paglia examined the fit of BPMN to the manufacturing domain. After the analysis they have proposed MPMN - an extension of BPMN by adding four elements that cannot be modelled explicitly with standard BPMN. The implementation is still considered challenging as the integration with existing systems requires a common understanding/definition of knowledge [28].

2.2.2 Ontology for Robotic Task Planning

Robots and robotic manipulators play a role of fundamental importance in the manufacturing industry, particularly for complex but repetitive tasks. However, the success of industrial robotics to date has depended strictly on a controlled, static environment [32]. The field of cognitive robotics deals with the development of robot systems equipped with awareness of their capacities and limitations in changing environments and agile production. To accomplish this, the development of an appropriate representation model for skills related to capabilities is necessary [33]. Therefore, the industry requires advancements in robotic systems to reduce the programming effort and to adapt to changing environments due to the new requirements of agile production.

Ontology plays a significant role in AI and robotics by providing structured data, reasoning, action understanding, context awareness, knowledge transfer, and semantic learning. The structured framework created by the ontology for knowledge representation is crucial for enabling intelligent behaviors in robots [6]. Ontology is a key enabler for AI in robotics. Ontology-based knowledge representation has been studied for its functionality in achieving robust autonomous robotic systems. When such a system is equipped with knowledge representation, the cognitive skills provided to the robot allow it to perform tasks, reason, and interact with various environments autonomously [6]. Ontology methods provide formalized concepts which may then be used to construct a knowledge base using classes, axioms and relationships. Thus, a shared domain vocabulary may be defined, making both collaboration and reusability simpler tasks [34]. Characteristics of ontology include formalization and standardization. When ontology is used for knowledge representation, it is easier and more flexible to share data, it makes reuse of knowledge possible, it provides clear structure and accurate data, and it supports ontology reasoning. Through this, knowledge-sharing among heterogeneous systems is made more convenient [9].

2.2.3 Ontology Background

Gruber's definition of ontology as "an explicit specification of a conceptualization" [35] has transformed to Guarino et al.'s "logical theory consisting of a set of formulas whose models approximates as well as (formally) possible the intended models, that is, those models that satisfy the conceptualization and the ontological commitments" [36]. Studer et al. stated that the ontology defined by Gruber characterizes best the essence of an ontology and they distinguished them between different ontological categories [37]. Ludger et al. [38] describe "an ontology [as] an artificial representation, that represents types or universals of a certain domain and the relations that hold according to a certain theory in a formal structure." In robotics and computer science, ontologies are considered "formal descriptions of objects, properties and relationships among objects collected in a particular data structure called the Knowledge Base (KB)" [7].

Based on the specific needs of the application of the ontology, ontology can be divided into different types [7]. Guarino et al. proposed four types of classification for ontology based on hierarchy levels: upper, reference, domain, and application ontology [7]. An upper ontology describes the world from a general way, and it captures cross domain widely applicable concepts. Consequently, these concepts are rather abstract and cannot be easily used for specific problems. Some well-known upper ontologies are the Suggested Upper Merged Ontology

(SUMO) [39], the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [40], and the IEEE standard Core Ontology for Robotics and Automation (CORA) [41]. A reference ontology is an ontology that focuses on a disciplined representation of deep knowledge of basic science with the goal of defining general terms and reusing these terms in different fields such as engineering, medicine etc. [5]. A domain ontology is limited to one area with the aim of describing the domain knowledge explicitly and formally [7]. The concepts and their relationships in this domain should be reusable and interchangeable across communities and industries. These concepts in domain ontology specialized the concepts in the upper ontologies and reference ontologies [42]. An application ontology focuses on modelling the knowledge of a specific application.

Olivares-Alarcos et al. have developed a more detailed classification of ontology based on ontology language and the ontology hierarchy [7]. Ontologies can be divided into informal and formal ontologies based on the ontology language [7]. The former signifies that the ontology is expressed without formal semantics for example, RDF (Resource Description Framework) [43], (part of) UML (Unified Modelling Language) [44], and BPMN (Business Process Model and Notation) [45]. The latter has associated formal semantics where the interpretation is described like FOL (First Oder Logic) and OWL (Web Ontology Language) which have clear and comprehensive semantic and syntactic rules [7]. This formal ontology can be further categorized into decidable and undecidable. Decidable ontologies such as OWL DL (Web Ontology Language Description Logic) can be used for information extraction at run time but undecidable ontologies like FOL and OWL are better at expressing and modelling knowledge [7].

2.3 Thesis Outline

This PhD thesis is divided into seven chapters: 1 Galician Summary, 2 Introduction, 3 Research Objectives, 4 Research Methodology, 5-8 Publications, 9 Results, 10 Conclusion, and 11 Bibliography. Chapter 5 to Chapter 8 describe the four publications in detail; each chapter is linked to one publication.

Chapter 5: Robot Process Modelling and Notation RTMN 1.0 is the first contribution of this PhD research. It is related to Publication 2 in Appendix 15.9. It covers the first rounds of literature review based on the research hypothesis and research questions to determine a gap in research. The goal of this work is to identify the research gap and propose a way to fill the gap. In this work a modelling language “robot task model and notation” – RTMN was developed to ease robotic process modelling and programming.

Chapter 6: Robot process modelling and notation RTMN 2.0 focus on Human robot collaboration is the second contribution of this PhD research. It is related to Publication 3 in Appendix 15.9. It is an extension of the former work on RTMN. It puts its focus on human robot collaboration. Notations related to human robot collaboration are created in this work. It combines the HRC safety modes and HRC task types and gives guidelines for HRC task modelling and execution.

Chapter 7: Ontology Support Robot Process Modelling is the third contribution of this PhD research. It is related to Publication 4 in Appendix 15.9. It concentrates on the creation of an ontology for the robot task management domain. This ontology is used to provide a standardization for the terms in this domain so that the terms can be reused and shared across different robotic systems. Another use of this ontology is to support the RTMN based graphical

user interface for knowledge retrieval and interface population. The last use of the ontology is to support the task planner with PDDL planning.

Chapter 8: Flexible Task Planner using RTMN2.0 and ORPP Ontology is the third contribution of the PhD research. It is related to Publication 1 in Appendix 15.9. It presents the ACROBA architecture focusing on the task planner. The task planner is the central components of ACROBA. It has several elements: the GUI (publication 1 & 2), the ontology (publication 3), skills manager and behavior tree generator etc. This work provides an overview of how different components work together as a whole and the synergy between the components as well as the contributions of the components.

The experimental validations for the implementation are explained in Chapter 9 Results.

Chapter 10 is the conclusion that summarizes the PhD thesis.

3 RESEARCH OBJECTIVES

Based on the technology background, potential areas for the PhD research field were identified:

- Connecting top level business processes with lower-level robot control
- Creating a common formalization for robotic task planning so that definitions like task, skill, primitive etc. are generalized from the different definitions in the literature, towards a standardization from process level to the robot control level.
- Developing an ontology enabled modelling language for robotic task planning and execution. This ontology-based task planning aims at providing a user-friendly tool allowing the user to compose robotic processes quickly and cheaply. The machine understandable feature of the ontology developed in this research serves as the common basis for communication between humans and machines by sharing the conceptualization of the robot domain.
- Developing a domain ontology that enables knowledge reuse and allows non-programmers to plan and program robot tasks in a flexible and easy manner.
- Creating an application ontology that covers light-out manufacturing and collaborative manufacturing in the context of the H2020 ACROBA project.

The main objectives and characteristics of this modelling language are:

- Provide an intuitive and easy to use graphical user interface (GUI) to design simulate and run the robotic processes.
- It defines the high-level symbolic planning for robotic tasks and creates the bridge to the low-level motion planning for robot control.
- Integration of human robot collaboration processes.
- Using “technology acceptance model” for technology design, development and validation.

The main objectives characteristics of this ontology are:

- It combines the tool-centric approach (a manufacturing task is described based on available hardware and software components) and the product centric approach (describing the product and associated production steps independent from specific production resources) [10].
- It includes upper-level ontology, domain-level ontology, and application-level ontology.
- It contains three kinds of knowledge in the domain ontology: process knowledge, task planning knowledge, and HRC knowledge. It integrates data from different sources, such as camera, sensors, domain knowledge and human input information into a single ontology [2].
- It achieves standardization, modularity, reusability, extensibility, knowledge sharing, and knowledge reasoning.

4 RESERCH METHODOLOGY

Initially, the author examined the different kinds of research methods. There exist many research methodologies one can adopt in this PhD study. After thoroughly comparing the advantages and disadvantages, a mixed method research was decided on - combining both quantitative and qualitative approaches (quantitative research uses surveys and questionnaires to gather numerical data through structured questions; qualitative research uses interviews that enables open-ended discussions to gather in depth information). The mixed methods research is combined with prototyping and testing. It provides several advantages which lead to a more comprehensive understanding of a research problem. The major advantages are as follows [11,12,13,46]:

- As it gathers both quantitative and qualitative data, it provides a more comprehensive understanding of the research question
- It not only covers the breadth of the research phenomenon, but also the depth
- It increases the credibility and reliability of the research by comparing and contrasting data from different methods
- It compensates for the limitations of each method. Qualitative data gives more context and depth to quantitative data while qualitative data offers statistical insights.
- It offers flexibility. One can use one research method to start a study and the other one to extend the study.
- This method answers not only “what”, but also “why” and “how” for the research questions.

Mixed research methods are used to design and validate the user experiences with the ontology and the tools implemented. In addition to the mixed methods research the author built prototypes to evaluate the functionalities and feasibility. Testing is used to improve system design and refine prototypes.

4.1 RESEARCH DESIGN

There are various mixed-methods research designs: convergent parallel design, explanatory sequential design, and exploratory sequential design. Figure 6 below provides an overview.

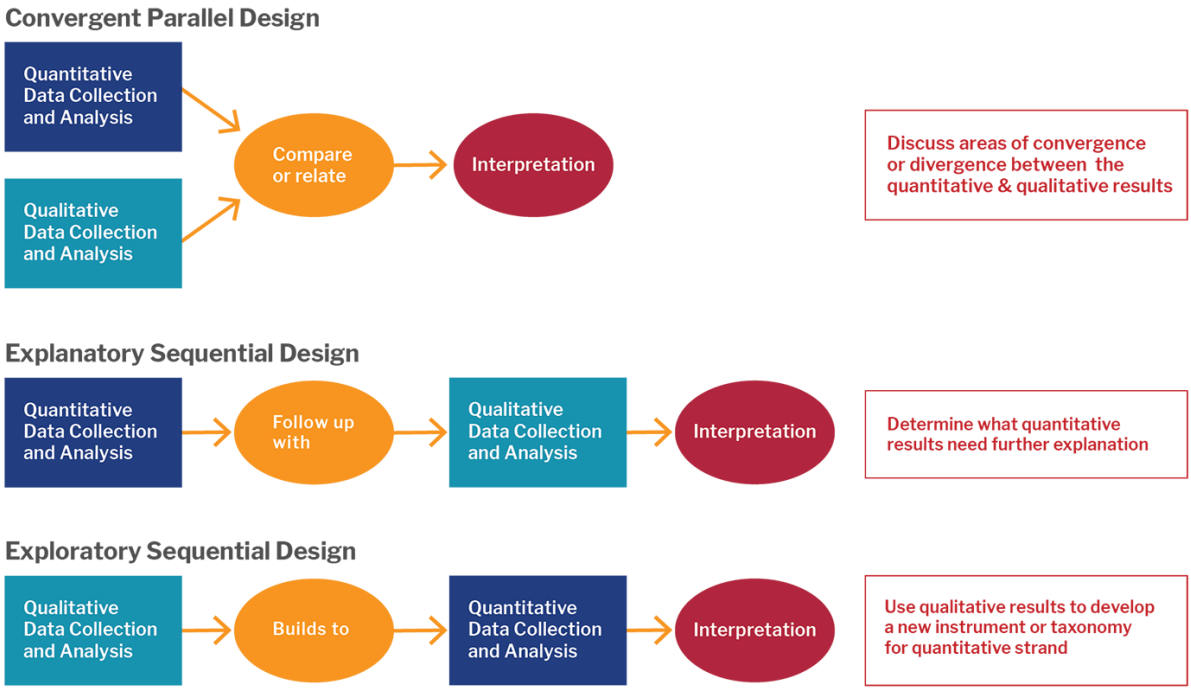


Figure 6. Basic Mixed Methods Research Design [45]

Figure 6 presents the basic mixed-methods research design. Convergent Parallel Design allows qualitative and qualitative data collection and analysis to be conducted in parallel. After this step, a comparison is run, and possible correlations are explored. The last step is interpretation of results. In this research design, it is important to have discussion of convergence or divergence between the qualitative and quantitative results. Explanatory Sequential Design carries out quantitative and qualitative data collection and analysis one after another. In this way, the quantitative results that need further explanation can be determined by the qualitative research method for further study. To establish an in-depth and more comprehensive study, the Exploratory Sequential Design method is selected for the research. The qualitative research method was used to build a detailed and structured study to gather the basic ideas of the users; and the qualitative research method was used to develop a new instrument or taxonomy for quantitative strand [47] based on interpretation of the quantitative study, which makes the quantitative analysis much more focused and comprehensive. To be more specific, questionnaires were used to collect the first round of data/requirements for the tool. Then based on the findings, the questions in specific areas could be refined using interviews as an instrument. This would allow the author to discover more details about the requirements. Based on these requirements derived from the quantitative and qualitative research, a first version of the tool (prototyping) would be built. Afterwards, the same approach would be used again to gather user feedback with questionnaires and then interviews. This round would produce further requirements which could be used to improve the tool and bring it to the next version. The overview is presented in Figure 4.

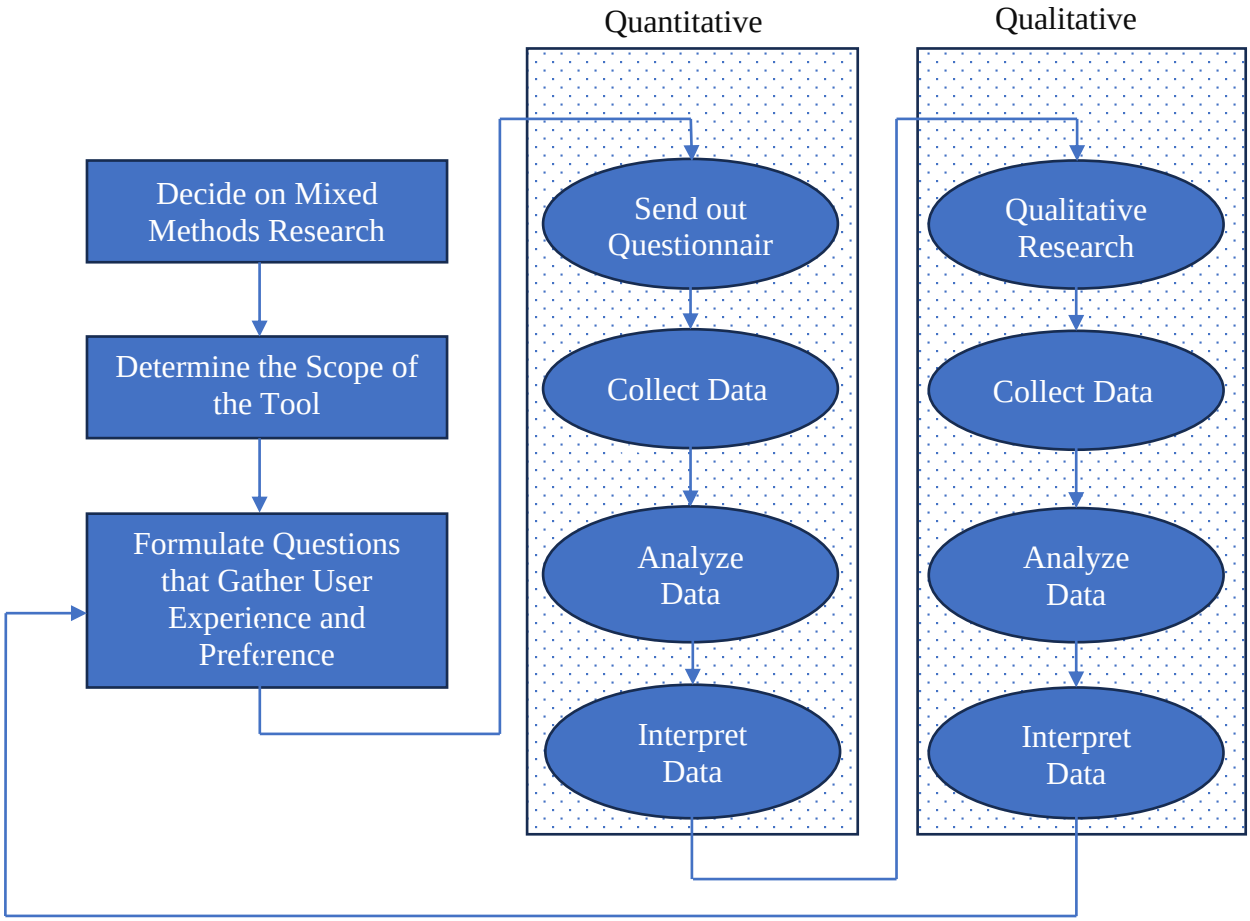


Figure 7. Explanatory Sequential Research Design

Figure 7 shows the research design. In the first step the general goal of the PhD research was identified. Based on that, the goals of the research questions were formulated. In the following step a mixed methods study was chosen as the research approach. Within the mixed methods research design, an explanatory sequential research design was decided. Quantitative research was carried out first to collect quantitative data. After analyzing the data, the first round of conclusions/statements were made. This was followed by qualitative research which used interviews as the main instrument. This then deepened the conclusions/statements drawn from the first round. The findings in this round were also used to improve the formulation of the questions that gather user experience and preferences.

5 ROBOT PROCESS MODELLING AND NOTATION RTMN1.0

5.1 INTRODUCTION

Flexible robot programming usually requires a lot of expertise. It is costly and has a long development cycle [1]. Therefore, it is very hard for the operators who manage the robots, or the process engineers who design the process to understand how to program a robot task due to a lack of high-level robotics expertise. There are frameworks and tools available to simplify robot programming. However, these tools are often too technical and not intuitive enough for operators and engineers [1]. They are also brand specific, with no possibility of changing from one supplier to another. To fill this gap, the author introduced an intuitive way of modelling and programming robotics tasks, which enables non-programming experts to model and program robot processes in an easier way. This publication describes an ontology-enabled skill-based robot task model and notation (RTMN) for planning and executing robotic processes, which is based on the BPMN (Business Process Model and Notation) [45] principles.

5.2 LITERATURE REVIEW

The literature review is divided into three parts (See Figure 8). In the first part, an overview of the problems operators have within robotic processes is provided. In the second part, existing graphical robotic task programming technologies and user interfaces is reviewed. In the third part, the author focuses on the BPMN [45] application in the robotics domain.

Figure 8. Literature Review Overview

5.2.1 Operators Facing Robotic Technology Challenges

5.2.1.1 Hard to Understand the Robots (error messages too technical, robotic processes too complex)

The number of industrial machinery and collaborative robots has increased drastically with Industry 4.0 [3]. This was followed by a high demand for flexible customization of production processes and automation workflows. As a consequence, end users in the industry are required

to be capable of programming industrial robots [3] McKinsey Global Institute's report also found that the skill set of workers needs to adapt to the increasing number of machines J. Wajcman, "New connections: social studies of science and technology and studies of work" [57]. However, Levine pointed out that workers usually cannot modify the new machines due to their complexity and high cost [3, p. 2]. Norman noted that on the one hand technology makes the life of workers easier with its increasing capabilities, but on the other hand its complexity also increases the worker's frustration towards technology [49]. A good system design allows the user to know the state of the system and possible actions one can take in a straightforward way. The user needs to know how the robot works and which information it relies on, as well as what it can do [49]. These are summarized to be the two usability issues: user's inability to determine the current state of the system, and the inability to determine the actions needed to perform a task.

When designing systems, user acceptance is paramount as otherwise the system will be avoided by the users [50]. Davis found that perceived ease of use and perceived usefulness were the two driving factors of the intention to use a system and actual system usage [15]. While many additional factors such as experience, age, gender, cost [50][15] have been identified which might influence these, the two above-mentioned factors remain at the core.

5.2.1.2 Reprogramming is Costly and Requires Expertise

Levine [3] points out that in the past, machines were complex and expensive, and workers were unable to modify the machines to complement their own skills. Jacob et al. emphasized that the worker's interaction with the machine should rely more on the understanding of the task to be performed than on the machine itself [53].

A report on a major accident in a chemical process plant clearly indicated that not all process control systems represent the state of the art in ergonomics [54]. Human factors are often neglected which makes the processes less controllable and [55] point out that when human operators are part of the system, process control design is then no longer simply a pure process control engineering problem.

The adaptability of humans is an important aspect of flexible manufacturing systems. Within automated assembly, the human workers have a crucial role which – essentially – comes down to their ability to solve unplanned problems safely [56].

It, therefore, seems logical that humans should be able to operate robots safely. For that to be possible, it is important that user interfaces are designed intuitively. The robot system should not only be simple to give inputs, but also provide basic information about the system so that it can facilitate intervention in unforeseen situations [23].

Robotics in modern factory environments is characterized by a need for higher levels of flexibility. Robot tasks need to frequently be adapted but the current setups require large amounts of time for programming [26].

Levine [3] points out that in the past, machines were complex and costly, and workers had few possibilities to modify the machines or to complement their own skills. Jacob *et al.* emphasized that the worker's interaction with the machine shall rely more on the understanding of the task to be performed than the machine itself [53].

Villani et al. wrote that when robots are working together with humans, the robot operation should be easy and intuitive under an ensured safe environment. They also pointed out the importance of the user interface and recommend designing the user interface to the convenience of the human operators and to accelerate interaction. To achieve these, they defined some

success-factors: 1. Enable easy and intuitive robot programming with practical input generation functions; 2. Provide adequate feedback from robots so that the users can gain situation awareness and use this information to understand the current system and robot behavior and be prepared for unforeseen cases and to react for dynamic planning [23]. Rovida et al. stated that to achieve the increasing degree of flexibility of modern robotics in the industrial environments, the effort and time required for programming is too high to allow frequent adaptations of robot tasks [26].

5.2.1.3 Hard to Accept Robot Technology

Research has shown that people's willingness to use a technology is linked to their perceived usefulness of the technology and their perceived ease of use [50][15]. Venkatesh and colleagues [52] discovered four factors that influence technology acceptance: age, gender, experience and voluntariness of use. [56] also pointed out that technology acceptance is linked to culture and gender, therefore, one needs to consider both design and use of the technology to increase the acceptance of the technology. As human workers still play an important role in flexible manufacturing, their adaptability and their safety should be considered for automation. As shown in the report [54] on the chemical process plant accident, not all process control systems represent the state of the art in ergonomics. This means the human factors are not considered important in the design of process control systems. This makes them less controllable. The human factors should be adequately defined and implemented and reflect the state of the art. F. Nachreiner, P. Nickel, and I. Meyer share the same opinion that when human operators are part of the system, process control design is then not any more simply a pure process control engineering problem [55].

5.2.2 Graphical Modelling/Programming Technologies

5.2.2.1 Visual Programming Languages (VPLs)

Visual Programming Languages (VPLs) provide a graphical interface to make programming easier. These VPLs are used for various purposes, ranging from children's games to robotic applications development to general-purpose programming [58]. Ahmad and Idrees pointed out that these VPL are good for teaching basic programming concepts, but they lack fundamental programming capabilities [58]. There are other user-oriented interfaces like the Task-Centered User Interface Design (TCUID) approach [20] or Measure, Adapt and Teach (MATE) framework [21].

I take one of these VPLs as an example to elaborate on. Google's Blockly is an open-source framework, which provides visual block-based icons and a drag-drop programming environment [59]. Winterer et al. [24] performed an expert review on the applicability of Blockly for industrial robot programming. They consider Blockly's puzzle-like, physical structure very good and highlight that Blockly's subroutines allow code re-usage within and across program boundaries. However, Blockly is not suitable for complex programs and expressions as understanding these subroutines is difficult for non-programmers [51]. In an attempt to add critical programming concepts - exception handling and debugging -which were

missing in Blockly, Ahmad and Idrees developed a Block-Based Visual Programming Language BBVPL [58].

5.2.2.2 Behavior Tree Language (BTL)

The Behavior Tree Language (BTL) has been proposed for complex engineering systems. It uses formal semantics (Formal semantics is the study of grammatical meaning in natural languages using formal tools from logic, mathematics and theoretical computer science) [60]. Colledanchise and Ögren [25] pointed out that the BTL has been very successful in the computer game AI community and is rapidly gaining popularity in robotics. [25] believe that modularity is the main reason for this as modularity is a key enabler when designing complex, maintainable and reusable systems. Due to the modular and adaptable representation of robotic tasks, BTLs can enable non-experts to program robots [25]. Furthermore, the enabled visual programming of BTL for end users can achieve the same level of complexity and power as traditional written programming code [25].

To fully exploit new technologies and learning algorithms, Villani et al. stated that for industrial applications it is necessary to develop programming interfaces [23]. Although there are many frameworks facilitating the development of robotics systems, high levels of expertise in programming are required [17]. Calling for simplification of robot programming is considered necessary [22].

5.2.3 BPMN Application in Robotics

The BPMN language is a standard maintained by the Object Management Group that can be used to model business processes [61]. A business process is “a defined set of business activities that represent the steps required to achieve a business objective. It includes the flow and use of information and resources” [61]. BPMN collaborations are highly expressive and encapsulate in a unique diagram the interplay among control-flow, dataflow, and communication in a flexible and understandable way, making the modelling activity easier. The resulting models are easy to read by humans and, at the same time, suitable for model-to-code solutions [17].

There are many works that aim to foster the use of BPMN. Sadik et al. [62] propose to use BPMN for modelling Multi-Robot System (MRS). Bourr et al. proposed an approach of disciplined use of BPMN 2.0 for high-level modelling of Multi-Robot Systems (MRS) [17]. BPMN is used to model a sequence of robotic activities in Traceable Robotic Activity Composer and Executive (TRACE) [30]. However, to capture domain specific features, extensions of the notation are often proposed [17]. There are extensions dealing with manufacturing processes [29], cyber-physical-production-systems [31], connection to smart technologies [63], and manufacture workflow [64]. Overall, BPMN diagrams are found to be highly expressive and easy for humans to read and simplify the modelling process [67]. But for smart manufacturing systems using IoT devices, despite the benefits of the open and standardized nature of BPMN, it is not adequate for modelling the physical aspects of a manufacturing system [63]. In the context of extending BPMN for Industry 4.0, [31] stated that modern workflow languages did not provide the structure, expressiveness, and flexibility to communicate with IoT devices, and the standard BPMN resource definition was described as

too abstract. Therefore, the conclusion can be drawn that standard BPMN is suitable for modelling business processes but needs extension to cover specific features for the robotic domain.

5.3 PROPOSED METHOD

Industry 4.0, which is characterized by smart manufacturing, denotes an evolutionary journey that aims at obtaining increased degrees of cooperation and communication in production systems [53] The idea of smart manufacturing is to create a network whose components have enhanced capabilities that can interact with each other in real time [29]. This has posed significant challenges to robotics process automation and modelling [65] [66]. According to Aspridou, there are many discussions about the use of Business Process Management to combine the different emerging technologies [29].

The literature review highlighted a gap between technology and users. On the one hand robotic implementation is growing rapidly, while on the other hand non-programming experts are facing more and more challenges when working with robots. The author believes this gap can be filled by adopting an intuitive and easy to use business process orientated graphical user interface at the front-end to model and plan robotic tasks and to utilize the more sophisticated programming technology and AI in the background to execute robotic tasks in ROS.

To further identify this gap and analyze the problem operators have, a mixed methods approach that combines both quantitative (questionnaires) and qualitative (interview) methods was adopted in the ACROBA research project [1]. For the research design the author followed the explanatory sequential design. The plan was to use questionnaires to collect the first round of data/requirements and - then based on the findings - to deepen the questions in specific areas using interviews as an instrument.

The questionnaires were created with Google Forms (see Figure 9). The complete questionnaires are presented in Appendix 15.2. These questionnaires were sent to 21 potential users of the ACROBA platform from five use cases (STERPACK, CABKA, MOSES, IKOR and ICPE [1]) to gather their requirements for the tool the author aimed to develop and discover the major problems they experienced when working with robots.

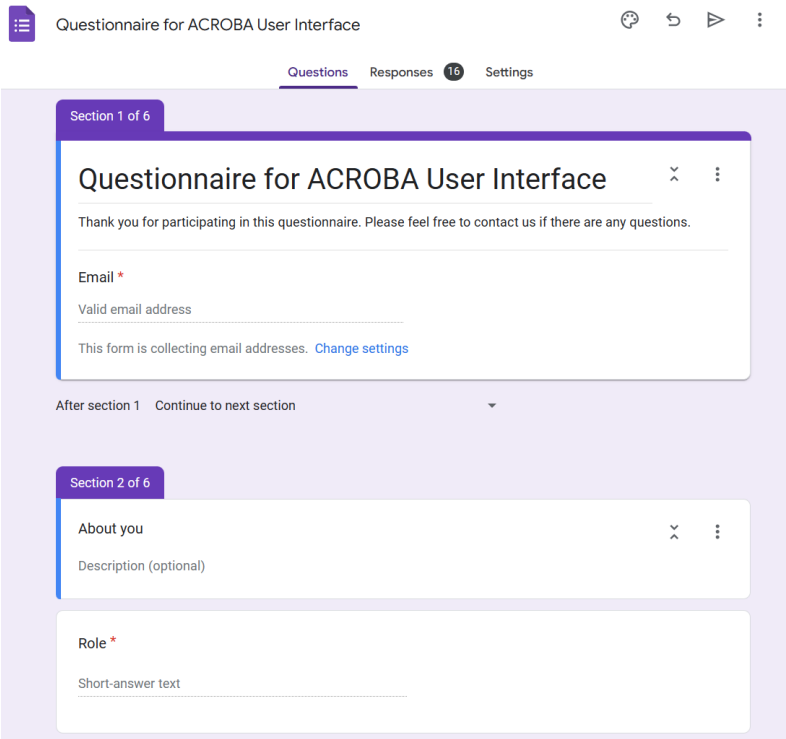


Figure 9. Google Form Questionnaires

Two weeks later all the inputs from the questionnaires had been collected. The full answers to the questionnaire are listed in Appendix 15.3. When analyzing the answers to the questionnaires, the author identified three core findings:

- There is a demand for an intuitive and easy to use interface to plan and control robot tasks.
- There is a need for users to determine the current state of the system, and to determine the actions needed to perform a task.
- There is a need for more information when an error occurs.

The key points are as follows:

- The two biggest problems users have when working with robots are that robots are hard to reprogram, and that the user interface is not user friendly.

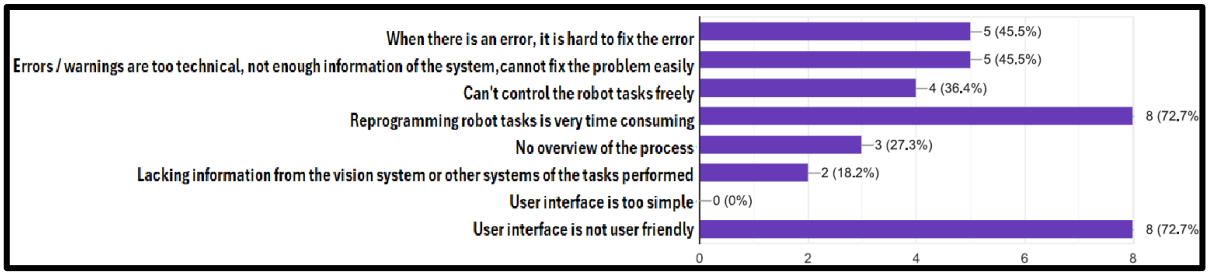


Figure 10. Question 11: What are the problems you have had working with robots?

- The most important features that the users expect for a user interface are “easy to use”, “graphical”.

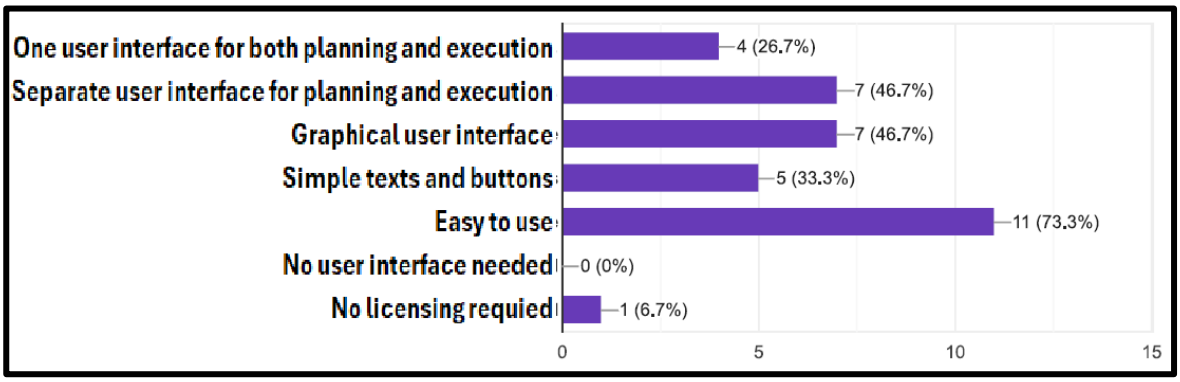


Figure 11. Question 4: What do you expect for a user interface of ACROBA?

- The robot task representation using BPMN basic elements (simple process flow chart) has the highest acceptance rate. It received 73.3% of the vote, which clearly won against other task representations like behaviors trees or Blockly. The reason is probably that users are very familiar with BPMN notations. Because they model their own process with similar diagrams using Viso or Word etc., they feel closer to this approach than other modelling notations.

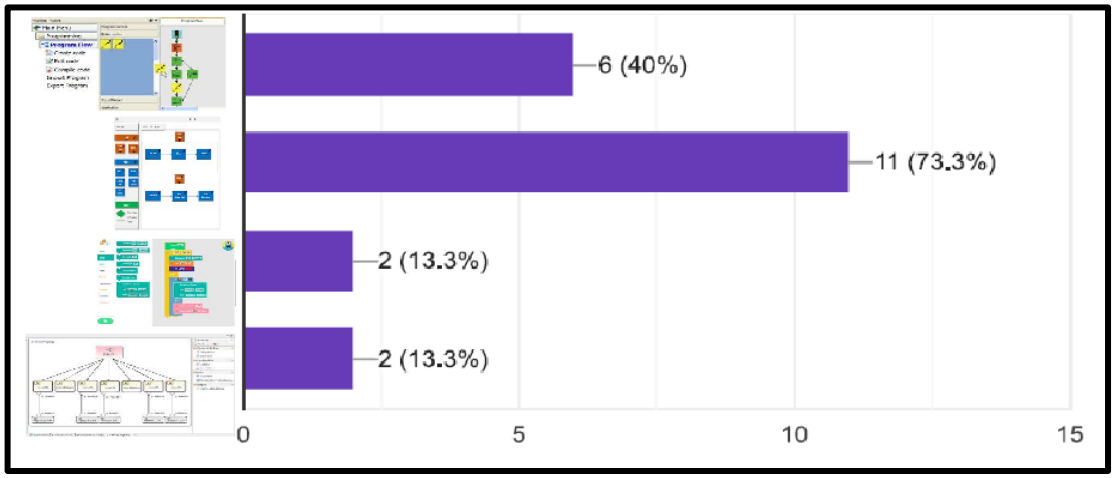


Figure 12. Question 5 Which of the following task representations do you like the most?

- When planning robot tasks, users prefer user friendly, drag and drop graphical user interface that can run simulation and optimization of tasks.

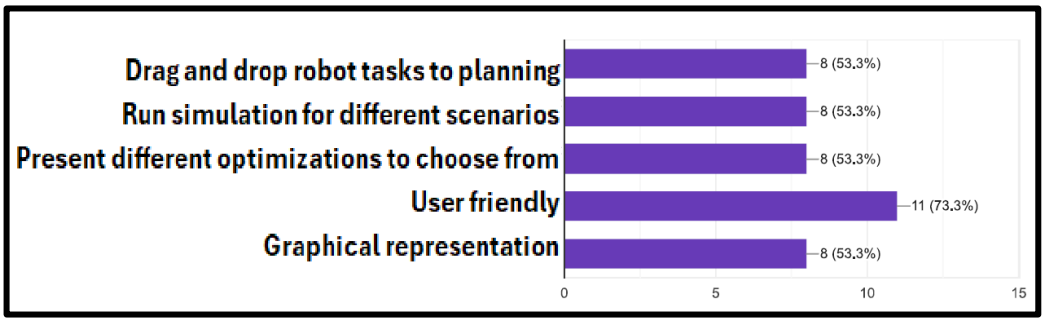


Figure 13. Question 6: What do you expect from a user interface for planning robot tasks?

The findings of the questionnaires match the gap identified in the literature. Based on all these inputs the author proposed a modelling language that combines modelling and programming in a graphical way for the robotic domain. The aim of this approach is to bridge robotic tasks modelling and programming in an easy way, which enhances usability and at the same time allows rich programming capabilities.

5.3.1 The Model

The name of the model is RTMN. It contains eleven basic notations, and they are differentiated by shapes, colors, and icons (see Figure 14). The idea is on the one hand to profit from the well-known business process modelling notation, and on the other hand to cover the robotics domain with robotics specific notation.

Notation	Notation Name	Attribute	Condition	Cardinality	Relation
	Human task	haveID haveName haveAgent (human performer) haveInput haveOutput (button/signal) haveLoopCharacteristics	No robot involvement	has at least 1 human performer	output is used to continue the process
	HRC Task	haveID haveName haveAgent (human performer) haveAgent (Robot) haveSequence haveInput haveOutput haveFeedback havePrecondition havePostcondition haveLoopCharacteristics haveDistance haveHumanPosition haveRobotPosition haveRobotSpeed	Human and robot are both involoved in completing the activity	has at least 1 human task and 1 robot task has at least one human input(signal/button)	has human task and robot task Four modes: Coexistence, Sequential work/Cooperation, Parallel work/Syncronised, Collaboration
	Robot Task	haveID haveName haveAgent (Robot) has skill haveSequence haveInput haveOutput haveFeedback havePrecondition havePostcondition haveLoopCharacteristics executeSkill executeTask	No human involvement	has at least 1 robot task	Has skills &/robot task. Used to compose other robot task and processes
	Robot Skill	haveID haveName haveAgent (Robot) haveSequence haveInput haveOutput haveFeedback havePrecondition havePostcondition haveLoopCharacteristics executePrimitive executeSkill executedbyRobottask	No human involvement	has at least 1 primitive	has primitives &/ skills Used to compose robot tasks has subclasses: generic skill, specific skill, saftey skill
	Skill Primitive	haveID haveName haveAgent(camera, sensor, scanner, gripper) haveInput haveOutput haveFeedback havePrecondition havePostcondition executedbyRobotskill	No human involvement	atomic, cannot be broken down to a finer level of detail	Receives information from other primitives and skills. Used to compose robot skills has subclasses:motion primitive, vision primitive, sensor primitive
	Data Flow	haveID haveName haveSourceDataRef haveTargetDataRef	Only used for data flow	has only one sourceDataRef and one targetDataRef	connecting data output and input
	Sequence Flow	haveID haveName haveSourceRef haveTargetRef	N/A	can connect only one element on the left and one on the right	connecting process elements
	Condition	haveID haveName haveGatewayDirection haveConditionSourceRef haveConditionTargetRefYes haveConditionTargetRefNo	N/A	at least can be split to two paths	is connected with sequence flows
	Start	haveID haveName	N/A	can only be used to start a process	can be connected to a sequence flow
	End	haveID haveName	N/A	can only be used to end a process	a sequence flow can be connected to it

Figure 14. Robotic Task Model and Notation (RTMN)

Four BPMN standard notations - “sequence flow”, “event” (start, end) and “gate way” (exclusive) are reused in RTMN. Definitions below come from the BPMN specification [67]:

“Sequence flow represents the sequence of Flow Objects in a Process or Choreography”.

“Start event: an Event that indicates where a particular Process starts. The Start Event starts the flow of the Process and does not have any incoming Sequence Flow but can have a Trigger. An end event indicates the end of a particular process”.

“Exclusive Gateway (Decision) is used to create alternative paths within a Process flow. This is basically the “diversion point in the road” for a Process. For a given instance of the Process, only one of the paths can be taken”.

“Inclusive Gateway (Inclusive Decision) can be used to create alternative but also parallel paths within a Process flow. Unlike the Exclusive Gateway, all condition Expressions are evaluated”.

There are five new notations: robot task, robot skill, robot primitive, HRC (human robot collaboration) task, and one slightly changed notation - human task, which is similar to a user task in BPMN, here it has a color assigned to it. In BPMN a “task” is an atomic activity within a process. A task is used when the work in the process is not broken down to a finer level of process detail [67]. However, in the robotics area the term “task” has a different usage. When engineers refer to a robot task - a picking task for example - it is not an atomic activity, it consists of several finer steps, such as to move to the right place, locate the gripper and grasp. Therefore, a robot task is not a typical BPMN task, it is rather a BPMN sub-process that consists of a set of activities. The author defines a robot task as a special type of sub-process, which only deals with robot activities. The activities within the robot task are defined as robot skills and skill primitives. A skill primitive is an atomic activity, and it is the lowest level a robot task.

5.3.2 The Implementation

This notation set is implemented as drag and drop elements in the graphical user interface (GUI) of the ACROBA platform. The design of the GUI is shown in Figure 15.

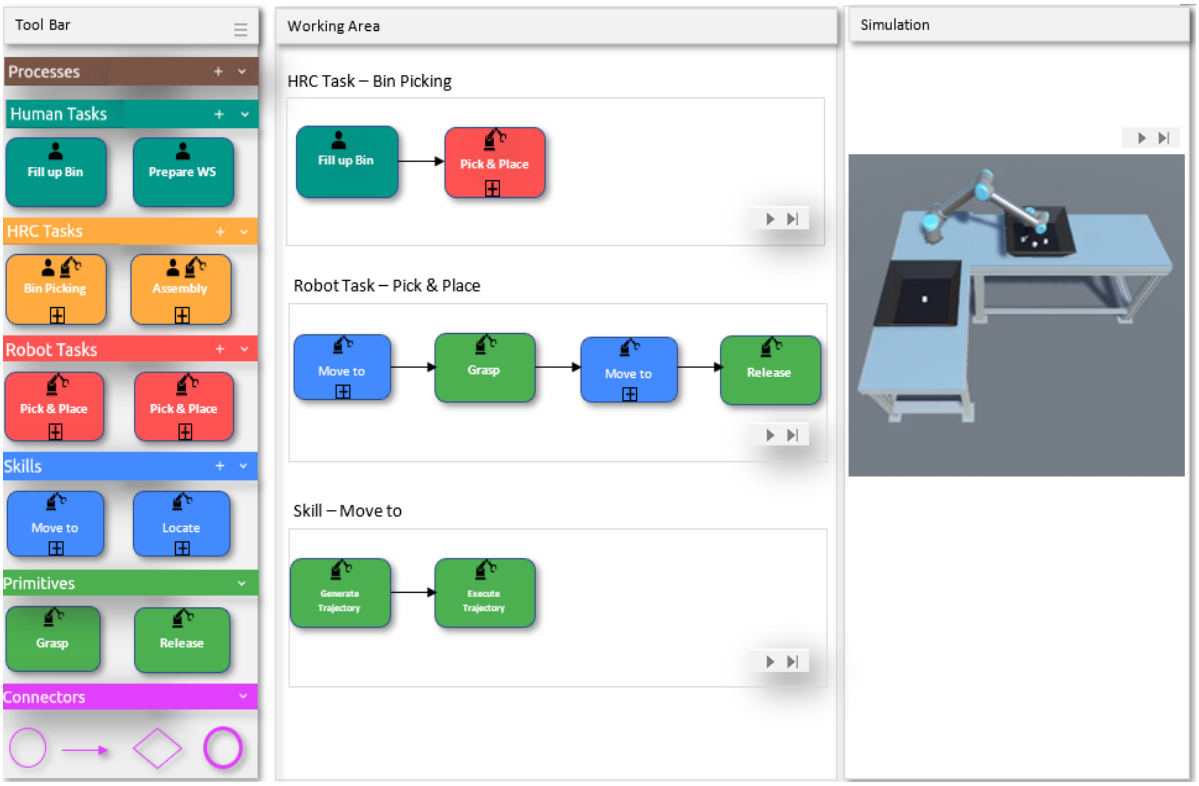


Figure 15. RTMN User Interface

The GUI is developed on the flutter framework [68], which is based on the dart programming language. Google Flutter is an open-source Flutter that allows you to build cross-platform apps from a single codebase that provides native performance [68]. It is chosen here because of its fast development cycle, expressiveness, and low costs.

Regarding the task design, the GUI will query information from the task planner directly from an ontology, where the skills are described, the primitives are available, as well as the description of all tasks modelled (sequence of skills used and their inputs) is given.

The use of ontologies for storing information of the state of the robotic platform and the tasks to be accomplished helps to implement algorithms that can automatically create or modify existing tasks, for example with the help of PDDL language. This approach is now widely used in the robotics research field with the development of several domain specific ontologies. An attempt at standardization was made by the working group ORA (Ontology for Robotics and Automation) of the IEEE-RAS in 2015 [69]. This core ontology has been extended in various research projects for industrial tasks [70].

Characteristics of ontologies include formalization and standardization. When Ontology is used for knowledge representation, knowledge becomes more accurate, structured, easier to share, and reasoning is supported. Thus, knowledge-sharing among heterogeneous systems and human-machine interaction is made more convenient [2]. Based on its benefits to the field of robotics, research was conducted to discover implementations of ontology and identify any gaps in the field. Through the research it was found that, although the use of ontology for task-planning in robotics applications has been thoroughly researched, its implementation in industry for the shift to mass customization has yet to see major advancements.

In ACROBA, the author combines and extends standardized ontologies with her own domain specifications and semantic rules, also taking inspiration from other existing ontologies. The ontology developed has a generic part which can be adopted by other applications and an ACROBA specific part that covers the five use cases to comply with the project semantics. The details of the ontology will be written in Chapter 7. The complete RTMN framework is shown in Figure 16.

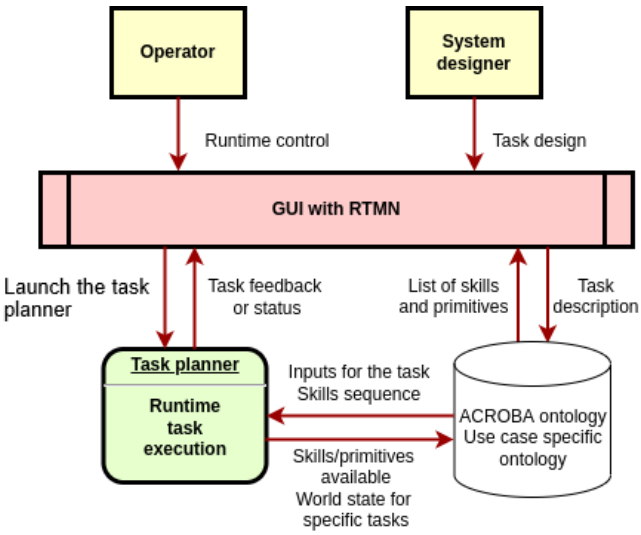


Figure 16. RTMN Framework

The system designer starts the task design through the RTMN user interface. The ontology provides a list of skills and primitives stored in it to the GUI for composing a task. After the task is composed, the GUI feeds the task definition and description back to the ontology to be saved as knowledge for future use. When the task is properly designed, the operator can run and control the task through the GUI. Once the operator presses start, the GUI will launch the task planner which is responsible for task execution. The task planner gets all relevant information related to the running task from the ontology and at the same time saves the information about the current instance back to the ontology.

5.4 VALIDATION

Based on Davis and Venkatesh, perceived ease of use and perceived usefulness of prototypes have been shown to predict the real use of the technology months later [71]. In order to develop a tool that the users will be using regularly and happy to use, Davis and Venkatesh's technology acceptance model was adopted.

After designing the GUI, the author made a presentation out of the prototype and made a video that was used to collect the perceived ease of use and perceived usefulness of the users. For this reason, a set of interview questions and a questionnaire was developed. The interviews are the second step in the explanatory sequential design of the mixed method research. These interviews are done with the users of the use cases of ACROBA. The purpose of these interviews is to deepen understanding of the requirements of the users. The questionnaire is used as an early validation of the tool.

The interview questions are shown below in Tabel 5:

Table 5. Interview Questions

What do you think about the GUI?
What would you suggest improving the GUI?
What are the roles of humans in your use case, what do they need from the GUI?
How do you want the errors to be displayed in the GUI?
Do processes need validation before being released? Who needs to validate the processes?

The detailed answers to the interview are presented in Appendix 15.4. Here is the summary of the main points:

- Require different levels of access
- High light the current task
- Allow customization
- Separate the skills/tasks/processes in progress and in released status
- Be able to program/compose new skills and tasks
- Process needs approval
- Simulation needed
- For errors a more detailed file is needed for users to fix the problem
- Manage parallel processes
- Color for human tasks should be improved

These interview answers were very valuable and will be used for further improvement and development of the tool.

The small questionnaire shown below in Figure 17 was used to collect feedback from users after seeing the presentation of the prototype. The answers are also presented in Figure 17. Besides the rated answers, some additional comments are provided by the users, this is presented in Appendix 15.5.

Prototype Validation					
Interview Questions	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
I find the naming of the user interface easy to understand	1	4	2		
I would like to use this system frequently		1	4	1	1
I find the symbles on the interface intuitive	1	5	1		
I think the system is simple and easy to use	1	5	1		
I think I will need technical support to use this system			4	3	
I find the system functions are sufficient			3	2	2
I think the drag and drop elements are well classified		7			
I think most people can learn this system quickly	2	3	1	1	
I find use of this system time-consuming			4	3	
I feel confident to use this system	1	1	5		
I think there are a lot of things to learn before I can start using this system	1	1	5		
I find the colors on the user interface comfortable and well chosen	1	3	3		

Figure 17. Question and Answers for Prototype Validation

Figure 18 shows the answers in a bar chart. One can clearly see that they perceived the GUI with RTMN as useful for managing their work and expressed their willingness to use the tool. Additionally, they found the design easy to understand and intuitive to use. Therefore, based on the technology acceptance models [15], the conclusion that the likelihood of a user to use the tool after implementation is high, and the acceptance of the tool is promising can be drawn.

Many additional requirements have arisen from the interviews. Human interaction with robots, user input, and parallel process became the major areas of focus for the next version of the GUI. These requirements were gathered and converted into new functional specifications for the development of the second version of the GUI. These functions will be described in chapter 6.

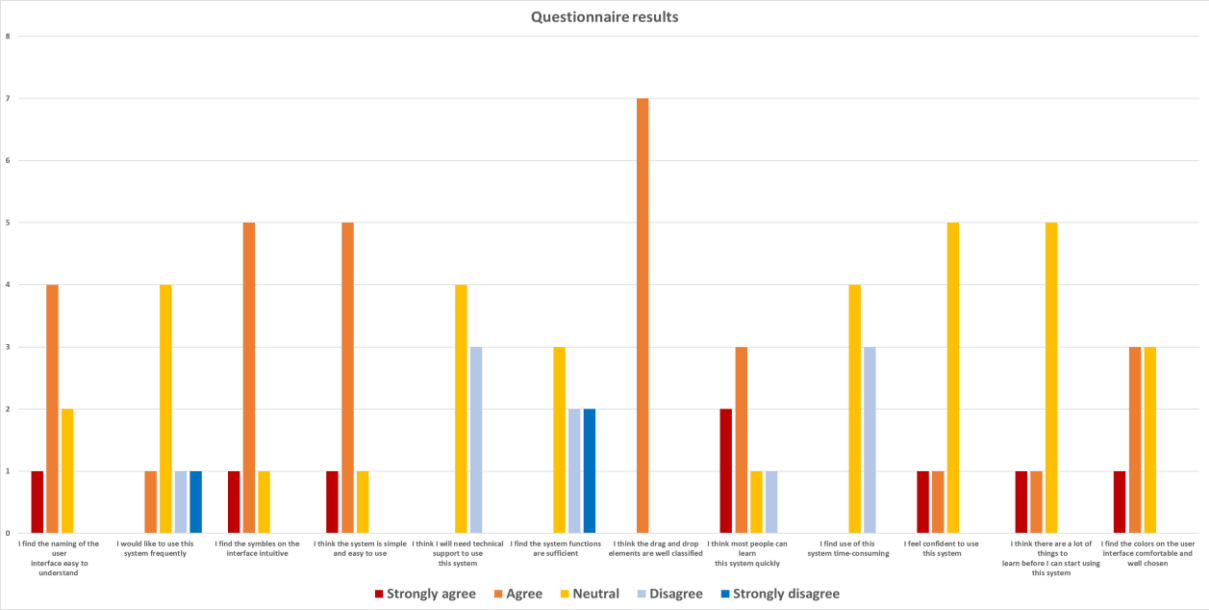


Figure 18. Answer Bar Chart (see Appendix 15.8 for the clearer version)

5.5 Conclusion and Outlook

This chapter covered the conceptual design of the RTMN and its implementation. RTMN aims to connect process modelling and robot execution. Reusability and intuitive robot programming allows people without programming expertise to plan and control robots. This chapter does not cover the details of modelling and executing human robot collaboration tasks. A comprehensive approach for modelling and executing human robot collaboration will be explained in chapter 6.

RTMN is implemented in the ACROBA project. The validation of RTMN is shown in Chapter 9.2 Scenario 1: Robot Programming through the GUI Using RTMN. This scenario shows the details of how RTMN is used to model a robotic process.

6 ROBOT PROCESS MODELLING AND NOTATION RTMN 2.0 FOCUS ON HUMAN ROBOT COLLABORATION

6.1 INTRODUCTION

As mentioned in Chapter 5, flexible robot programming is usually costly and requires much expertise. The development cycle often lasts months or years [1]. This puts significant pressure on operators and engineers. On the one hand, they are required to control the robot, and on the other hand, it is difficult to understand the robot's tasks due to a lack of high-level robotics expertise. Although frameworks and tools are available to simplify robot programming, these tools often are not flexible (brand-specific), are too technical, and are not intuitive enough for operators and engineers [1].

In Chapter 5, the author developed a modelling language called RTMN (Robot Task Modelling and Notation) [72]. It aims to ease the modelling of the robotic process. The basic elements are based on Business Process Modelling and Notation (BPMN) [45]. The main motivation for developing RTMN was the challenges that non-robotics experts are facing in the fast-growing agile production industry: cost and requirement of robot programming expertise, as well as knowledge of the complexity and difficulties of controlling robotic systems. The validation results indicate that users find RTMN notations simple to understand and intuitive to use.

This research work covers the second generation of the RTMN. RTMN 2.0 adds many additional notations: four human–robot collaboration (HRC) tasks, requirements, and KPIs; condition checks and decision making; join/split; and data association. HRC is the focus among these notations. This research is also conducted under the research project ACROBA.

6.2 LITERATURE REVIEW

6.2.1 What is Human–Robot Collaboration (HRC)?

The ever-growing advancements in the manufacturing industry have made it necessary for companies to optimize their systems by decreasing human workload, fatigue risk, and overall costs. This necessity has led to the introduction of human–robot collaboration in industrial environments [73]. Industry experts have established that the complete removal of humans from manufacturing systems is not viable. A more realistic goal is to have humans and intelligent machines working in harmony [74]. From an anthropological standpoint, "collaboration" refers to multiple parties communicating with each other and coordinating their actions in order to achieve a common goal. To this aim, collaborators observe each other, infer the intent behind an action, and plan their own actions in accordance with this intent. Similarly, in human–robot collaboration, robot systems should be designed with the appropriate tools to coordinate their actions with humans and employ the relevant cognitive and communicative mechanisms such that they can plan actions toward an established goal [75].

For the successful implementation of human–robot collaboration, the machines would require advanced cognitive capabilities to allow the human operators to collaborate comfortably

and efficiently and maintain high confidence in these systems. If implemented correctly, industries may achieve a reasonable task reduction for human operators. To this end, robots should be equipped with understanding capabilities that allow them to operate with a human, just as two humans would when working together [76]. HRC systems do not rely on an equal divide of workload between humans and robots. The levels of robot automation are based on the application and are decided such that they lead to an overall improvement in the system's performance [77]. This improvement in performance can be attributed to the complementary strengths of either party. Robots offer efficient and guaranteed performance at high speeds, whilst humans offer understanding, reasoning, and problem solving [73].

6.2.2 The Importance of Human–Robot Collaboration (HRC)

The aim of industrial robotics is to enable efficient performance repeatedly and accurately [78]. Currently, assembly lines have important requirements related to adaptability, mainly due to the rapid rate at which new products are introduced to the market, as well as changing technologies. The current trends in industry reflect a shift from "mass-production" to "mass-customization". Products now come with numerous variants or upgrades and a much shorter product lifetime. This imposes a challenge regarding flexibility and adaptability on the manufacturing process, a challenge for which human–robot collaboration is an attractive arrangement [79]. Companies have traditionally relied on robots with in-built capabilities and limited flexibility. However, this level of flexibility is not enough to match the current market's demands [80]. While traditional robot systems provide high payload capabilities and repeatability, they then suffer from limited flexibility and dexterity [81]. Thus, human–robot collaboration is a suitable arrangement to leverage the unique capabilities of both humans and robots for increased efficiency and quality in industrial scenarios. The recent trend of automation and data exchange, known as Industry 4.0, also supports the use of collaborative systems in industry. The aim of Industry 4.0 is to achieve efficiency, cost reduction, and increased productivity by means of integrated automation. This aim highlights the need for flexible and interoperable systems, including intelligent decision-making software and robots that can be quickly, safely, and intuitively operated by humans [23].

Industries are increasingly relying on HRC arrangements, both from an engineering perspective and from a socio-economic standpoint. While the manufacturing industry is a significant source of employment, it has been reported that most jobs offered by this sector may remain unfilled. This is attributed to a shortage of workers with the relevant technological and technical skills [82]. HRC is, therefore, a promising alternative, which makes up for the skill gap, still requires human operators, and may be more attractive to the younger generation. Additionally, robotic systems result in higher competitiveness with countries with cheap labor systems and increase trust in the company's technological aptitude. Collaborative systems also alleviate the ergonomic burden on human workers, resulting in an improved work environment and a reduction in occupational injuries. This makes environments that include both robots and human laborers more attractive to interested partners, customers, and the public [83]. Modern technologies of intuitive systems such as augmented reality, walkthrough programming, and programming by demonstration are all simple methods to operate collaborative robots, unlike the advanced technical expertise necessary to operate traditional robotic systems [23].

Currently, collaborative robotic solutions are attractive even to small- and medium-sized companies since such systems are more affordable, compact, and easy to use compared with traditional robotic systems. Traditionally, factory floors have had strict divisions of labor, with

robots confined to strict safety cages far from humans. Collaborative robots overcome this division of labor, allowing humans and robots to work closely together. In doing so, the advantage of automation provided by the robot is combined with the flexibility and intuitive nature of the human [23,81]. Evidently, there are numerous advantages to collaborative robotic systems, including economic, social, and ergonomic improvements to traditional systems. However, to harness the full benefits of such systems, companies should adhere to the appropriate safety standards to ensure optimal operation. These will be discussed in the following section.

6.2.3 Safety Standards and HRC Modes

The International Federation of Robotics has reported an all-time high of 517,385 new industrial robots installed in 2021 in factories around the world, with a growth rate of 31%, increased by 22% compared to 2018 (pre-pandemic record). Until now, the stock of operational robots around the globe has reached 3.5 million units [84]. With the increasing use of robots in industry, standardization, and guidelines to ensure the safety of human operators are required [79]. Many standards have been proposed to give guidelines for the safe use of collaborative robots. Machinery safety is regulated under the Machinery Directive, which covers the scope of collaborative applications [79]. The following reference standards are reported (see Table 6).

Table 6. Safety Standards

Type	Description	Standard
Type A Standard	Basic safety standards for general requirements	ISO 12100 [85]: 2010 “Machine safety, general design principles, risk assessment, and risk reduction” IEC 61508: terminology and methodology
Type B Standard	Generic safety standards	B1 standards (ISO 13849-1[86], IEC 62061 [91]): specific safety aspects B2 standards (ISO 13850 [87], ISO 13851 [88]): safeguard
Type C Standard	Safety countermeasures for specific machinery Prioritized over Type A and Type B standards	ISO 10218: safety of industrial robots ISO 10218-1 [89]:2011 “Robots and equipment for robots, Safety requirements for industrial robots, Part 1: Robots”, safety requirements for robot manufacturers (robot and controller) ISO 10218-2 [90]: 2011 “Robots and equipment for robots, Safety requirements for industrial robots, Part 2: Systems and integration of robots”, safety requirements for system integrators (robot and ancillary devices), ISO TS 15066 [92]: 2016 “Robots and robotic devices, Collaborative Robots”, guidance on collaborative robot Operations

Four categories of safety requirements are defined for collaborative robots in the type C international standards (ISO 10218-1, ISO 10218-2, and ISO TS 15066) [23,79,81]:

6.2.3.1 Safety-rated Monitored Stop (SMS)

SMS [23,79] is a collaboration arrangement in which robot motion is stopped before a human operator enters the collaborative workspace to interact and carry out a task with the robotic system. This is the most basic form of collaboration and takes place within a collaborative area, that is, an area of operation shared by robots and humans. Both parties can

work in this area, but not simultaneously since the robot cannot move if the operator is in the shared space. Therefore, it is ideal for tasks in which the robot primarily works alone and is occasionally interrupted by a human operator. Examples of such tasks include visual inspection or the positioning of heavy components by the robot for the human.

6.2.3.2 Hand-Guiding (HG)

Another mode of collaboration is known as hand-guiding [23,79], or "direct teach". In this mode, the operator simply moves the robot to teach it, without the use of intermediate interfaces such as teach pendants. These positions are communicated as commands to the robot system. Throughout this process, the robot arm's weight is compensated such that its position is held. A guiding, hand-operated device is used by the operator to guide the robot's motion. For this advanced form of collaboration, the robot must be equipped with safety-rated monitored stop and speed functionalities. Once the robot has learned the motion and the human operator has left the collaborative area, the robot may execute the program in automatic mode. However, if the operator enters the area, the program is interrupted. When the operator is using the hand-guiding device, the robot operates in a state of safety-rated monitored speed functionality ("safety-rated function that causes a protective stop when either the Cartesian speed of a point relative to the robot flange, or the speed of one or more axes exceeds a specified limit value"[21]) until the operator releases the arm and leaves the collaborative area, allowing the robot to resume automatic operation once again.

6.2.3.3 Speed and Separation Monitoring (SSM)

In SSM [23,79], the robot operates even when a human is present by means of safety-rated monitoring sensors. Thus, human and robot operations take place simultaneously. To reduce risks, a stipulated protective distance must always be kept between the two parties. If this distance is not kept, the robot operation stops and only resumes once the operator has moved away from the system. If the robot system operates at a reduced speed, the protective distance is reduced accordingly. The workspace may be divided into "zones", whereby if the human is in the green zone, the robot may operate at full speed, if in the yellow zone, the robot operates with reduced speed, and if the human enters the red zone, the robot's operation is stopped. Vision systems are used to monitor these zones.

6.2.3.4 Power and Force Limiting (PFL)

PFL [23,79] is a collaborative approach in which limits are set for motor power and force such that the human operator and robot may work side-by-side. These limits are set as a risk reduction method, defined by a risk assessment. To implement this approach, specific equipment and control modes are required to handle collisions between the robot and human and prevent any injuries to the human.

These four collaborative modes can be applied to both traditional industrial robots and collaborative robots. For traditional industrial robots, additional safety devices such as laser sensors or light curtains are required. On the other hand, for collaborative additional features

such as force and torque sensors, force limits, vision systems, laser systems, and anti-collision systems are required [81].

6.2.4 HRC Task Types

It is important to analyze the different types of collaboration tasks [23,79,81]. Matheson et al. used the classification that Müller et al. [93] proposed for human–robot collaboration in their paper [81], which distinguishes HRC task types into four groups: coexistence (same environment, no interaction), synchronized (same workspace, different times), cooperation (same workspace, at the same time, separate tasks), and collaboration (same workspace, same task, same time). Wang et al. [94] presented the following types in their paper: coexistence (not sharing workspace, no direct contact), interaction (sharing workspace, communicating with each other, performing tasks sequentially), cooperation (sharing workspace, having individual goals, sharing resources, working simultaneously), and collaboration (joint activity, sharing workspace, having the same goal, physical contact allowed). Thiemermann [95] differentiated four operating modes: manual mode (human), automation (robot), parallelization (same product, direct contact, suitable for pre-assembly), and collaboration (same product, work together). There are other classifications in the literature [96-98]. To summarize, there are four basic task types for HRC based on the literature: coexistence, sequential cooperation, parallel cooperation, and collaboration.

6.2.5 HRC Modelling Methods

Choosing the appropriate modelling language is imperative for the development of a robust human–robot collaboration system. This choice depends on various factors, including the particular task requirements, the target platform or framework, and the specific preferences and technical skills of the developers. Commonly used modelling languages in this field include Business Process Modelling Notation (BPMN), Systems Modelling Language (SysML), Behavior Trees (BTs), Unified Modelling Language (UML), and Petri Nets. Literature related to each of these languages is reviewed and discussed in this section.

6.2.5.1 Business Process Modelling Notation (BPMN)

BPMN was originally developed as a standard notation for business users to bridge the gap between business process design and process implementation. A Business Process Model contains a network of graphical objects representing activities or work and the flow controls representing the order of performance [99]. While BPMN is typically used for business processes, its capabilities can be extended to the field of robotics. More specifically, it can be used to model the flow, tasks, decisions, and interactions that occur in HRC processes.

One approach found that there was a lack of consideration for the modelling of collaborative tasks and therefore attempted to completely model processes such as assembly workflows, interactions, and decisions made by machines and humans. It supports variability modelling, making it attractive for customized manufacturing scenarios. A BPMN-based workflow designer was implemented to allow the user to model processes intuitively [64].

BPMN was also used for the development of a risk analysis software tool to support changes in adaptive collaborative robotic scenarios [101]. The work presents a tool that can automatically identify any changes that have been made to a particular application's components or processes, providing safety experts with a tool to monitor manufacturing processes. The results showed that this technology provided better usability and decreased errors when compared with conventional methods. BPMN was used as a basis for development, with certain modifications to make it appropriate for collaborative scenarios. However, this application is only used for change management related to risk analysis.

Another paper tackles the task of warehouse material handling using an automated guide vehicle system [102]. The Manufacturing Process Management System (MPMS) is used to control the process. The aim is to achieve a system that performs all tasks automatically, without requiring major changes for various processes. The MPMS was designed using BPMN and Camunda as a platform, allowing for the BPMN model to be executed and controlled in real time.

Concerning the lack of efficiency arising from static task allocation, one paper focused on the use of adaptive task sharing in manufacturing and assembly [103]. A model was developed for experimental purposes, which included task sharing worker assistance software based on BPMN. The results indicated that adaptive task sharing reduced the productivity gap in automated assembly.

6.2.5.2 Unified Modelling Language (UML)

UML is a popular modelling language in the fields of software engineering and system design. Through its standardized notation, clear visualization and specification of structures, behaviors, and interactions of systems is achieved. UML is, therefore, thought to be an ideal approach to model architecture, interfaces, and interactions in HRC systems. The use of UML in the field of human–robot collaboration seems to mostly focus on safety and risk analysis. This can be seen in [104-107].

F. D. Von Borstel et al. [108] used a combination of UML and Colored Petri Nets (CPNs) to model and simulate mobile robots based on Wireless Robotic Components (WRCs). UML diagrams were customized to describe specific WRC architecture based on robotic software, particular tasks, and operational environment. The hierarchical CPNs were developed using customized UML diagrams as a guide. From there, an executable model of robotic system could be developed. Further work will aim to improve the translation from UML to CPNs. This approach was developed for mobile robots which navigate and work autonomously, this is very different from industrial and collaborative robots where autonomy is not the focus.

L. Carroll et al. [109] attempted to use UML in development for the design of real-time robot controllers, based on the necessity of better adaptability from robots when working in cooperation with humans. The main focuses were dependability, fault avoidance, fault removal, and fault tolerance. UML was found to be appropriate for the development and maintainability of this controller.

As a major software engineering modelling tool, UML can model requirements and high-level system architecture very efficiently. However, UML is not well-established for robotics, including HRC. While UML can be useful for modelling certain aspects of human–robot collaboration, limitations arise when applied to the full scope collaborative scenarios. Robots often operate autonomously and need to make decisions, but UML does not offer specialized modelling constructions for representing autonomy or decision-making processes.

6.2.5.3 Systems Modelling Language (SysML)

SysML is a standardized formal language, of which the main purpose is the modelling of systems engineering processes. It is based on UML but is more well-adapted for complex systems in engineering and automation.

Researchers proposed SysML as a modelling language as it was thought to simplify robot programming through abstraction [110]. Since robots are increasingly being integrated into environments where humans are present, they should be made safer, with straightforward programming and interaction capabilities. To achieve this, SysML was used to represent manufacturing processes by graphical diagrams capturing the system's structure from various perspectives. While this work did not focus directly on modelling human–robot collaboration, it was one of the motives for the approach, and the capabilities of the system show promise to be adapted for HRC scenarios.

While the literature specifically focusing on using SysML for HRC tasks is limited, SysML has been widely used for modelling general robotic workflows. K. Ohara et al. [111] proposed robot software design based on SysML. The language was chosen due to its benefits in terms of reusability and flexibility. R. Candell et al. [112] aimed at achieving real-time observation and control using SysMl modelling for architecture, components, and information flows. This was based on Industry 4.0 and the resultant requirements for adaptability, flexibility, and responsive communication. SysML is used to develop a graphical model for a wireless factory work cell.

In summary, although SysML can be a useful tool for modelling certain aspects of HRC such as requirements, system architecture, and interaction with external systems, SysML is not the most suitable tool for modelling HRC due to its focus on technical aspects (HRC involves not only technical aspects, but also human factors which are not the focus of SysML), abstraction level (SysML is normally for modelling system architecture, structure etc. at a high level of abstraction, while HRC needs to model social interactions, human behaviors which are at lower level of abstraction), lack of explicit support for modelling human behaviors and social interactions as well as support for dynamic and adaptive modelling.

6.2.5.4 Behavior Trees

Behavior trees have been widely used in the field of robotics to define the behavior of autonomous agents. The graphical modelling language involves a hierarchical structure of tasks and dependencies, making them an ideal approach to model complex decisions and coordination in HRC scenarios.

The literature related to behavior trees is largely concerned with the limitations of automation to capture human dexterity and judgement, especially in unstructured environments where unforeseen events commonly occur. A framework was thus presented, which captured various behaviors through which robot autonomy is enhanced. This was achieved using behavior trees to model the robot's intelligence, such as social behaviors, human intention, and various tasks, including collaborative ones [113].

While behavior trees can be a valuable tool for modelling the autonomous behavior of a robotic system, they are less suited for modelling the complexities of human–robot collaboration that involve, for example, communication, shared decision-making, and social interaction. Behavior trees tend to focus on predefined actions and decision sequences, making them less flexible in terms of accommodating the varied and sometimes unpredictable nature

of human intention. Behavior trees prioritize system-centric decision-making and are not adept at handling human-centric decision processes. The design of behavior trees is not human-centered, nor user-friendly; this is another drawback of behavior tree modelling.

6.2.5.5 Petri Nets

Petri nets were developed primarily for the modelling of systems in which events may occur concurrently, with constraints on the concurrence [114]. Petri nets for human–robot collaboration was found in the literature. A. Casalino et al. [115] presented a scheduling method for collaborative tasks in assembly, which aims at achieving optimal planning and adaptability of assembly process based on runtime knowledge. Similar work was carried out in refs. [116,117]. Colored petri nets were used to model the concurrent scenario in which a human works collaboratively with a wearable robot. The developed model assigned tasks to the robot that were considered too laborious for the human in a panel installation task [118]. The main aim of the work carried out by R. E. Yagoda et al. [119] was to develop a technique to model Human Robot Interaction (HRI). A. Casalino et al. [120] demonstrated this in a realistic scenario involving a human operator and dual-arm robot performing an assembly task using time petri nets. However, the method is case-specific and needs to be extended to adapt to general cases in future research. Petri nets were found to be advantageous, as they are generally simple yet are built upon underlying mathematics, making them powerful for such modelling applications. Additionally, their graphical nature makes them a great communication medium.

Petri nets have limitations on expressiveness for complex decision making, task allocation, and communication. Their design is mainly used for discrete event modelling other than real-time and continuous motion and perception. Petri nets also do not naturally integrate with AI and learning algorithms. They are not considered the most suitable choice for modelling the complexity and dynamics of human–robot collaboration, which involves a combination of continuous and discrete interactions, uncertainty, and complex decision-making processes.

6.2.6 Research Gap

To summarize, UML and SysML are more well-adapted for high-level system modelling, requirement, and system integration. They are more geared for technical people instead of systems' end users. Behavior trees then mainly focus on automation, predefined actions, and decision sequences, which are not suitable for humans nor HRC. Petri nets lack expressiveness for decision making and communication. Although BPMN notations have been found to be well-accepted by its intuitive modelling notation in terms of usability, the standard BPMN focuses on business process modelling and does not have separate notations for robots (robot tasks, robot skills, robot primitives, and HRC tasks). Although BPMN can be applied to robotics, the process model becomes too complicated for non-experts in BPMN. In general, there is a need to model HRC, but these aforementioned modelling tools are not suitable for this purpose. They are all well-established tools in their application areas. However, they are not developed specifically for robotics in the HRC domain. In the current versions of these tools, it is not possible to separate the HRC task types, specify the HRC modes, verify safety standards, nor monitor human factors.

For the abovementioned reasons a new version of RTMN to extend the RTMN to cover the human robot collaboration area was proposed. The author named this new version RTMN 2.0. In the next sections, this shall be explained in more detail.

6.3 Proposed Method

In this section, RTMN 2.0 – the extension of RTMN will be described in detail. First, the former RTMN elements and then the additional elements in RTMN 2.0 will be introduced. Afterwards, the RTMN 2.0 sequence flow connection rules are presented. The HRC model follows this and is the core contribution of this paper. Later, other extensions and modifications of RTMN are added. Finally, the implementation and demonstration of RTMN 2.0 are described at the end of this chapter.

6.3.1 The RTMN Elements

RTMN originally contained eleven basic notations differentiated by shapes, colors, and icons [72]. These notations were developed based on the well-known business process modelling notation (BPMN) [45]. BPMN is commonly used to model business processes. The notations in BPMN are very familiar to both business and technical users. To model a process BPMN notations are the most suitable, however, to model the robotic actions there are no standard notations in BPMN. In order to provide a familiar and easy-to-use modelling environment, the basic well accepted BPMN notations are used instead of reinventing the wheel. To cover the robotics domain new notations were developed. These notations adopt colors, icons, and shapes in their design, aiming to make the modelling more intuitive. Four BPMN standard notations, “sequence flow”, “event” (start, end), and “gate way” (exclusive), are reused in RTMN (see BPMN specification [45]).

There are five new robotics-specific notations: robot task, robot skill, robot primitive, HRC (human–robot collaboration) task, and one slightly changed notation—human task. It is like a user task in BPMN; however, here, it has a human icon, and a color assigned to it. The reason to separate robot task, human task and HRC task is because the agent who perform the tasks are different – robot task is performed by robot, human task by human and HRC task by both. A “task” in BPMN is the lowest level of the process and cannot be broken down into finer details. It is considered as an atomic activity within a process [45]. However, the term “task” in the robotics area is, rather, a subprocess in relation to BPMN because in BPMN task is atomic. A robot picking task, for example—it is not an atomic activity but consists of detailed steps such as to move to the right place, locate the gripper and grasp, etc. Therefore, a robot task is differentiated from a typical BPMN task. The author defines it as an activity consisting of a set of activities that corresponds to a special type of BPMN sub-process that deals with robot activities. The activities within the robot task are defined as skills. Primitives are atomic activities that compose robotic skills. Skills form robot tasks. The concept of skill and primitive are based on the model of Pedersen et al. [137]. The advantage of introducing skills is “a skill-equipped robot can react to changing products and production volumes - the key point of implementing transformable solutions in robotics” [137].

6.3.2 RTMN 2.0 Elements

RTMN 2.0, in comparison to the first version of RTMN, has the following extensions:

Adding HRC modelling elements, including safety in combination with collaboration modes and task types of humans and robots. This has significantly enriched the former RTMN model and enabled it to be applied to HRC application areas.

- Adding requirements, with KPI as a basic element.
- Adding the link from requirements/KPI to robot control.
- Adding decision-making elements

RTMN 2.0 was designed especially for modelling human–robot collaboration processes. Together with version 1.0, it is capable of modelling light-out automated robotic processes as well as human–robot collaboration processes. The model contains two sets of elements: the RTMN elements and the extended elements. A complete overview is presented in Figure 19-21. It consists of all the elements of RTMN 2.0, with the newest improvements and extensions.

Notation	Notation Name	Description	Attribute	Condition	Cardinality	Relation
	Process	Process is a complex activity or set of activities that accomplish a specific organizational goal. It is a orange (#ED7D31) double line rectangle with round corners	has name has KPI has requirement has goal has task has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition	N/A	has at least one task	a process can be connected to another process
	Human task	It is an activity that is performed by a human It is a green (#009688) rounded corner rectangle with a single thin line and includes a human figure marker in the top middle of the rectangle	has name has human performer has inputs has output (button/signal)	no robot involvement	has at least 1 human performer	
	Robot Task	Robot task is a compound activity that done by robots to perform a complex robot action. It can be broken down into finer level of details. It is a red (#FF5252) rounded corner rectangle with a single thin line and includes a robot figure marker in the top middle of the rectangle, a “plus” sign in the lower-center of the shape indicating that the activity has a lower level of details.	has name has skill has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition	no human involvement	no human operator involved.	has skills. Used to compose HRC tasks and processes
	Skill	Robot skills are used to integrate and syncHRConize robot actions and sensor data in a consistent way. A robot skill is an elementary functionality or action provided by a robot in order to perform an autonomous mission. It is a blue (#448AFF) rounded corner rectangle with a single thin line and includes a robot figure marker in the top middle of the rectangle, a “plus” sign in the lower-center of the shape indicating that the activity has a lower level of details.	has name has skill has primitive has sequenceElement has inputs has outputs has feedback has precondition has postcondition	no human involvement	has at least 1 primitive	has primitives &/ skills Used to compose robot tasks
	Primitive	Primitives are simple, atomic robot activities that can be combined to form more complex robot behaviors. They are the lowest level model elements and provide direct access to hardware elements. It is a green (#4CAF50) rounded corner rectangle with a single thin line and includes a robot figure marker in the top middle of the rectangle	has name has inputs has outputs has feedback has precondition has postcondition	no human involvement	atomic, cannot be broken down to a finer level of detail	receives information from other primitives and skills. Used to compose robot skills

Figure 19. RTMN modelling elements—Part A: Basic Notations

A process consists of tasks, these tasks can be human tasks, robot tasks, or human robot collaboration tasks. Human tasks are atomic, and they do not have a finer level of details. Robot Task can be composed of robot skills. Robot skills are composed of primitive and other skills. This is different from Pederson’s model where skill can be only made by primitives. The reason

for that is to enable more flexibility on robot programming. Not limiting this will allow all skills and primitives available for programming a new skill.

When a human task is finished and robot needs to continue, the human needs to tell the system that his/her work is done. This can be configured based on the requirements. For example, it can be given by click on the “continue” button on the GUI or press a predefined physical button that gives feedback to the system that indicate the human work is finished so that the robot can continue it work. When a robot finishes its work and a human needs to continue, the human needs to be informed. This can be configured as email notifications or popup windows on the GUI depends on the requirements.

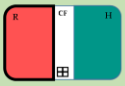
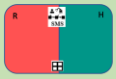
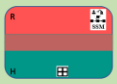
	HRC Task - Coexistance Fence (CF)	Human robot task CF is a compound activity that is performed by both human and robot. It can be broken down into finer level of details. This HRC task type is a combination of coexistance task type and a minimum HRC mode Fence. It is a round corner rectangle with a single thin line. It has two colours the red (#FF5252) represents robot and the green (#009688) represents human. It has a white space in between indicates the robot and human do not work at the same workplace. The thick black line indicates robot in fence. It has a "plus" sign in the lower-center of the shape indicating the activity has a lowerlevel of detail.	has name has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition has loopCharacteristics has HRCTaskType has HRCMode has Safety Level			
	HRC Task - Sequential Cooperation SMS(SS)	Human robot task SS is a compound activity that is performed by both human and robot to achieve a common goal. It can be broken down into finer level of details. This HRC task type is a combination of sequential cooperation task type and HRC mode SMS. It is a round corner rectangle with a single thin line. It has two colours the red (#FF5252) represents robot and the green (#009688) represents human. It has an icon indicating the sequential cooperation task type and HRC mode SMS. It has a "plus" sign in the lower-center of the shape indicating the activity has a lowerlevel of detail.	has name has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition has loopCharacteristics has HRCTaskType has HRCMode has Safety Level	human and robot are both involved in completing the activity	has at least 1 human task and 1 robot task	has human task and robot task has HRC task type: sequential has HRC mode: SMS has Safety level: >=1
	HRC Task - Teaching HG(TH)	Human robot task TH is a compound activity that performed by both human and robot to achieve a common goal. It can be broken down into finer level of details. This HRC task type is a combination of teaching task type and HRC mode HG. The middle brown area with the joined color of red and green represents the shared workspace by human and robot. It is a round corner rectangle with a single thin line. It has two colours the red (#FF5252) represents robot and the green (#009688) represents human. It has an icon indicating the collaboration task type and HRC mode. It has a "plus" sign in the lower-center of the shape indicating that the activity has a lowerlevel of detail.	has name has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition has loopCharacteristics has HRCTaskType has HRCMode has Safety Level	human and robot are both involved in completing the activity	has at least 1 human task and 1 robot task	has human task and robot task has HRC task type: teaching has HRC mode: HG has Safety level: >=2
	HRC Task - Parallel Cooperation SSM(PS)	Human robot task PS is a compound activity that performed by both human and robot to achieve a common goal. It can be broken down into finer level of details. This HRC task type is a combination of parallel cooperation task type and HRC mode SSM. The middle brown area with the joined color of red and green represents the shared workspace by human and robot. It is a round corner rectangle with a single thin line. It has two colours the red (#FF5252) represents robot and the green (#009688) represents human. It has an icon indicating the collaboration task type and HRC mode. It has a "plus" sign in the lower-center of the shape indicating that the activity has a lowerlevel of detail.	has name has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition has loopCharacteristics has HRCTaskType has HRCMode has Safety Level	human and robot are both involved in completing the activity	has at least 1 human task and 1 robot task	has human task and robot task has HRC task type: parallel has HRC mode: SSM has Safety level: >=3
	HRC Task - Collaboration PFL(CP)	Human robot task CP is a compound activity that performed by both human and robot to achieve a common goal. It can be broken down into finer level of details. This HRC task type is a combination of collaboration task type and HRC mode PFL. It is a round corner rectangle with a single thin line. It has two colours the red (#FF5252) represents robot and the green (#009688) represents human. It has an icon indicating the sequential collaboration task type and HRC mode SMS. It has a "plus" sign in the lower-center of the shape indicating that the activity has a lowerlevel of detail.	has name has sequenceElement has inputs (goals) has outputs (results) has feedback has precondition has postcondition has loopCharacteristics has HRCTaskType has HRCMode has Safety Level	human and robot are both involved in completing the activity	has at least 1 human task and 1 robot task	has human task and robot task has HRC task type: joint has HRC mode: PFL has Safety level: >=4

Figure 20. RTMN modelling elements—Part B: HRC tasks

These five notations are explained in detail in Section 6.3.4.1. The HRC tasks require at least one robot and one human to complete the task. One HRC task can have several human tasks and robot tasks in it. All these tasks are done together to complete the HRC task. For example, in assembly, a human operator is in charge of preparing the components for assembly and a robot picks and places the components, the human operator does the correction if robot

place something wrongly. These then involve two human tasks: preparing components and error correction, one robot task: assembling components.

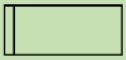
	Requirement	Requirements are KPI targets to achieve certain process goals.	has name has requirementDes has goal has KPI has formula has input has output has precondition has postcondition	N/A	at least one	connecting to process or subprocess
	KPI	KPI stands for key performance indicator, a quantifiable measure of performance over time for a specific objective.	has name has KPIDes has formular	N/A	quantitative KPI, at least one	connecting to requirement
	Sequence Flow	connect flow objects in a process, the arrow direction indicates the workflow direction	has name has sourceRef has targetRef	N/A	can connect only one element on the left and one on the right	connecting process elements
	Data Association	connect data input or output to a flow object in a process.	has name has sourceRef has targetRef	N/A	can connect only one element on the left and one on the right	connecting process elements
	Condition	Condition is rule-based decision making. It is enabled by a rule table where users can specify their decision logic. It is a dimond shape with a rule table icon.	has name has ruleTable has alternativePath	Rule table	system can execute automatically	is connected with multiple sequence flows
	Decision	Decision refers to human decision making. Human decision making allows operators/users to specify a question that will be answered at run time. It is a dimond shape with a human icon.	has name has condition has alternativePath has humanInput	Decision table	Human needs to decide here by giving input	is connected with multiple sequence flows
	Workspace	Workspace is an area in the robotic cell/workplace. It is used to seperate the different physical spaces. It is a black rectangle with a smaller rectangle in it on the left.	has name has inputs has outputs has feedback has precondition has postcondition			
	Start	indicating where a process start	has name	N/A	can only be used to start a process	can be connected to a sequence flow
	End	indicating where a process will end.	has name	N/A	can only be used to end a process	a sequence flow can be connected to it

Figure 21. RTMN modelling elements—Part C: other notations

The basic elements were explained in Chapter 5. The additional elements are explained later in this chapter. There is a difference between sequence flow and data flow. Both are used to connect process elements. But the former is the main connector to connect most of the elements in the process. It indicates the direction of the time flow of the process. Data association indicate only belongings and has associations. For example, a process can have requirement(s) and a requirement can have KPI(s).

6.3.3 RTMN 2.0 Sequence Flow Connection Rules

I display the rules regarding how to connect the elements in a sequence flow in Figure 22. The basic rules are as follows: “Start” can only start a process and cannot have any element flow into it. “End” is the opposite; it ends the process and cannot have any elements flow out of it. “Sequence Flow” connects the other elements and points out the direction of the process flow. Requirements are normally defined on the process level, but they can also be assigned to tasks, skills, and primitives if needed. KPIs are usually used to measure the achievement of requirements for the process but can also be assigned to tasks, skills, and primitives if needed. Tasks can be connected to any other task, condition, decision, or sequence flow. Skills and primitives can be connected to skills and primitives as well as conditions, decisions, and sequence flows.

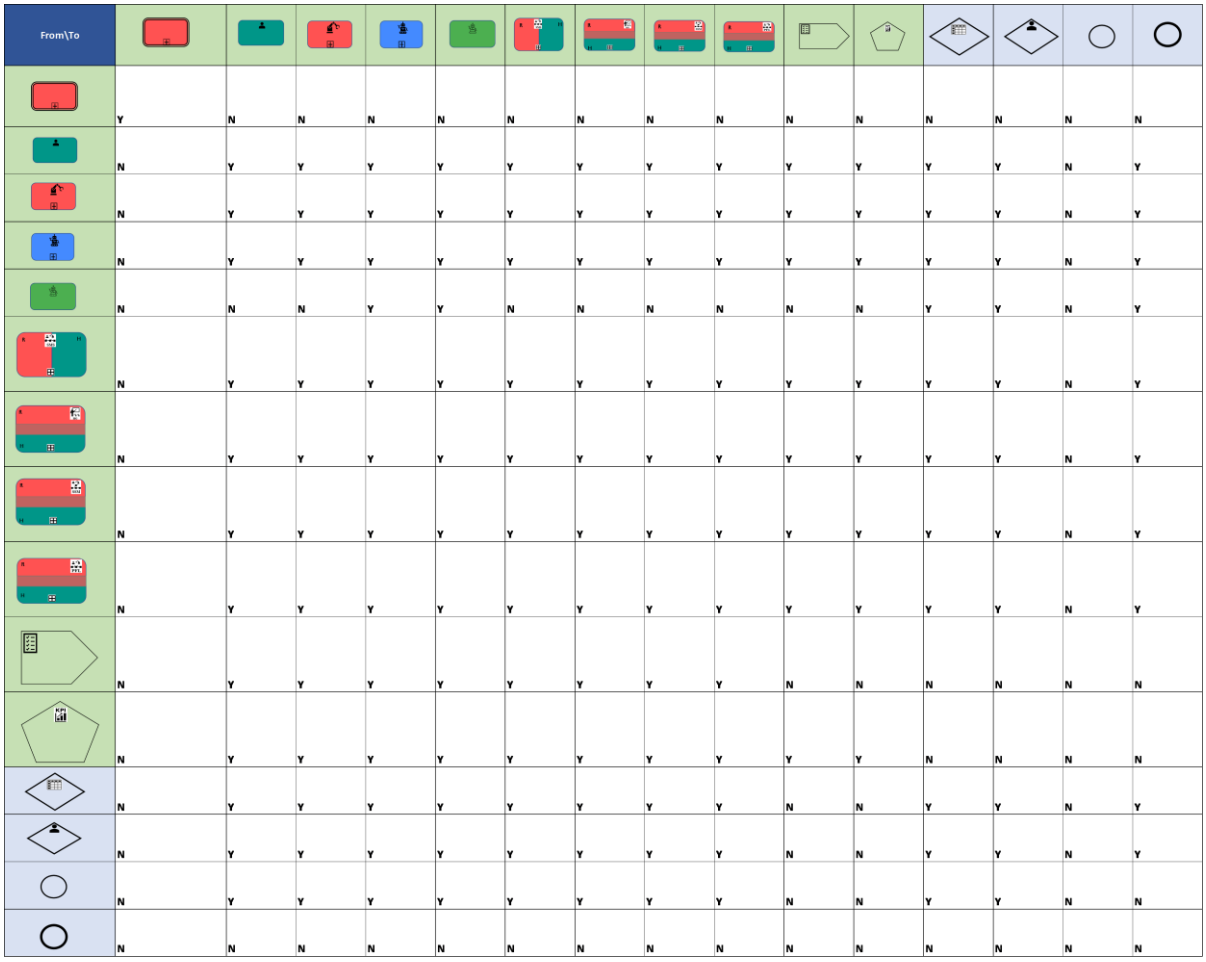


Figure 22. RTMN 2.0 sequence flow connection rules

6.3.4 The HRC Model

The author has developed an HRC model that covers the following aspects of HRC: HRC task types, HRC modes, combining the HRC task types and HRC modes, workspace, and decision making. The author proposed hardware and software solutions for each HRC task type. Each aspect is addressed one by one in the following sections.

6.3.4.1 Combining Collaboration Task Types and HRC Modes

Based on the literature ([23,79–98]), the different classifications of the HRC task types and HRC modes are consolidated. In the literature, five HRC modes (fence, SMS, HG, SSM, and PFL) and five HRC task types (cell, coexistence, synchronized, cooperation, and collaboration) are commonly defined. The HRC modes can ensure the safety of people even when unexpected human intrusions occur. However, the HRC task types do not by definition ensure the safety of people. This classification focuses on how closely humans and robots work together, and only the people planned for the task are considered. In reality, no matter how the task types are defined, humans can act differently than expected, and human intrusions can occur. Therefore,

to ensure safety, each task type was required to have a minimum safety mode enabled. For this reason, the author presents a new classification that integrates the HRC task types, HRC modes, and safety levels. Safety level 0 indicates no risk to the human from the robot. It has the lowest safety requirements, as the robot is within a physical fence. The safety levels go up to level 4, which requires the highest safety requirements not only from hardware but also from software. The author also distinguishes the collaboration levels (0–5): the higher, the more collaboration between human and robot.

For each task, general safety requirements were defined. A solution design for the hardware and software level is proposed for each HRC task type. This makes it easier for process engineers to ensure safety when implementing HRC processes. The overall approach is presented in Figure 23.

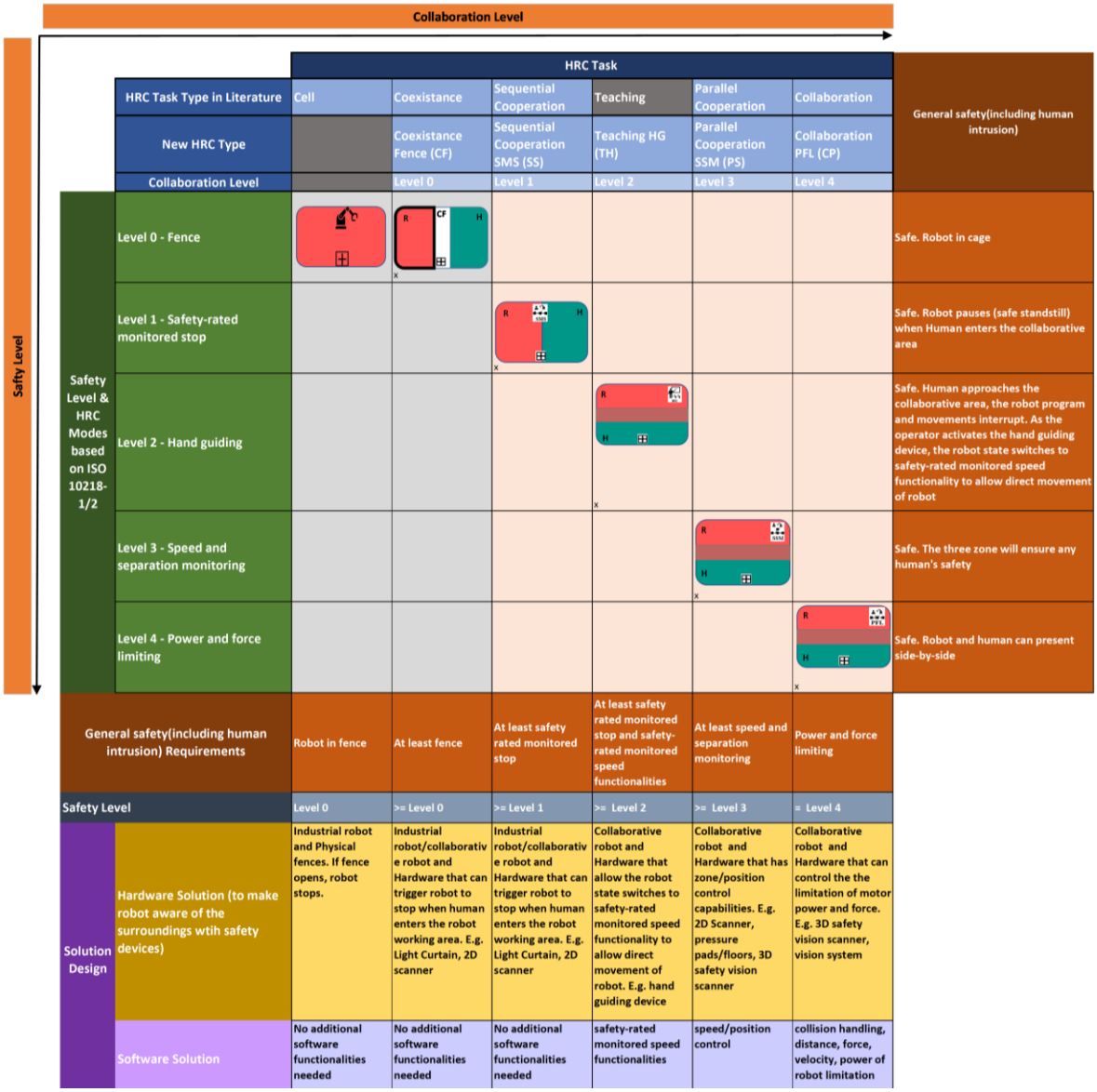


Figure 23. Combining HRC task types and HRC modes

In the author's new classification, there are five categories: Coexistence Fence (CF), Sequential cooperation SMS (SS), Teaching HG (TH), Parallel cooperation SSM (PS), and

Collaboration PFL (CP). The "cell" of the human robot task type refers to the robot working autonomously, surrounded by physical fences without human involvement, so only the robot tasks need to be modelled, and no human tasks need to be modelled. In this case, a separate HR task type was not necessary for it, as it has only robot tasks. The author added a new task type, "Teaching", which is when a human takes control of the robot and demonstrates what the robot should do, then the robot learns and repeats it. The reason for adding it is to cover this missing collaboration task type and then to be able to map it to the collaboration mode HG.

6.3.4.1.1 Coexistence Fence (CF)

This type (Figure 24) combines the task type "coexistence" and the collaboration mode "fence". This task has the lowest HRC level 0 and should activate at least safety level 0 (fence) in the background. Using an industrial robot is recommended, since humans and robots do not work together in the same workplace. The fence for the robot can be in the form of a physical barrier or certified hardware such as a light curtain/2D scanner. As long as fence mode is insured, this task is safe. No additional software functionalities are needed.

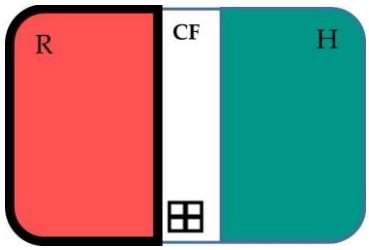


Figure 24. Coexistence Fence Notation

6.3.4.1.2 Sequential Cooperation SMS (SS)

This type (Figure 25) combines the task type "synchronized" and collaboration mode "safety-rated monitored stop". "Safety-rated monitored stop" means "condition where the robot is stopped with drive power active, while a monitoring system with a specified sufficient safety performance ensures that the robot does not move", as described in [122].

This task has collaboration level 1 and should activate at least safety level 1 (SMS) in the background. Using either an industrial robot or a collaborative robot with safety-certified hardware is recommonded, such as a light curtain/2D scanner to ensure SMS (robot stops when human enters the robot working area). As long as SMS mode is ensured, this task is safe. No additional software functionalities are needed.

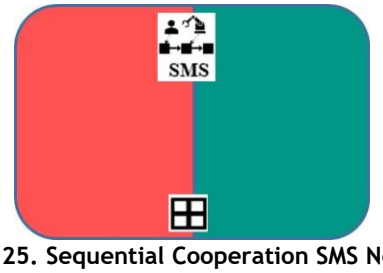


Figure 25. Sequential Cooperation SMS Notation

6.3.4.1.3 Teaching HG (TH)

This type (Figure 26) combines the task type "teaching" and collaboration mode "Hand Guiding". This task has collaboration level 2 and should activate at least safety level 2 (HG) in the background. Using a collaborative robot with safety-certified hardware that allows the robot's state to switch to safety-rated monitored speed functionality is recommended to allow for the direct movement of the robot, e.g., a hand guiding device. If a minimum safety HG mode is ensured, this task is safe. Safety-rated monitored speed functionalities need to be programmed at the software level. In ISO 10218-1, safety-rated monitored speed is written as "safety-rated function that causes a protective stop when either the Cartesian speed of a point relative to the robot flange, or the speed of one or more axes exceeds a specified limit value" [122]. In this mode, both robots and humans are presented at the same time. In the hand guide mode, a human holds the robot arm and guides it to learn for example a trajectory. The robot is not moving by itself. It is passively moved by the human.

Figure 26. Teaching HG Notation

6.3.4.1.4 Parallel Cooperation SSM (PS)

This type (Figure 27) combines the task type "parallel cooperation" and the collaboration mode "speed and separation monitoring". This task has collaboration level 3; it should activate at least safety level 3 (SSM) in the background and put in place the hardware that ensures the SSM mode. Using a collaborative robot with safety-certified hardware that has zone/position control capabilities, e.g., a 2D scanner, pressure pads/floors, or a 3D safety vision scanner is recommended. Speed/position control functionalities need to be programmed at the software level. The work area should be divided into three zones: green zone (robot runs at full speed), yellow zone (robot runs at reduced speed), and red zone (robot stops). As long as SSM mode is ensured, this task is safe.

Figure 27. Parallel Cooperation SSM Notation

6.3.4.1.5 Collaboration PFL (CP)

This type (Figure 28) combines the task type "collaboration" and the collaboration mode "power and force limiting". This task has collaboration level 4 and should activate safety level 4 (power and force limiting) in the background. This type is the most collaborative task type, requires the highest safety controls, and demands the most software engineering effort. Using a collaborative robot with safety-certified hardware that can control the limitations of motor power and force, e.g., a 3D safety vision scanner is recommended. For force control and limitation, force/torque sensors in end-effectors and/or in the joints of the robot are required. At the software level, functionalities need to be developed for collision handling, distance monitoring, force control, velocity tracking, and power of robot limitation. As long as PFL mode is ensured, this task is safe. For this notation, one must assign HRC mode 4 and task type collaboration to it.

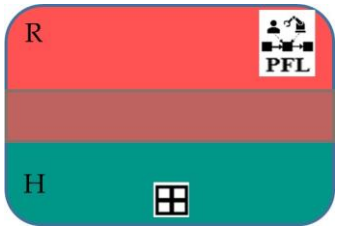


Figure 28. Collaboration PFL Notation

For both Parallel Cooperation SSM (PS) and Collaboration PFL (CP) the velocity needs to be monitored. In addition to the 2D/3D scanner, on the software level more speed and separation monitoring functions can be added. The minimum separation distance between the robot and the human (D_{HR}) with the following parameters: Speed Robot (V_R), Speed Human (V_H), Reaction Time (T_R), and Stop Time (T_S). An illustration of the formula is shown in Figure 29.

- $D_{HR} = V_R * (T_R + T_S) + V_H * (T_R + T_S)$

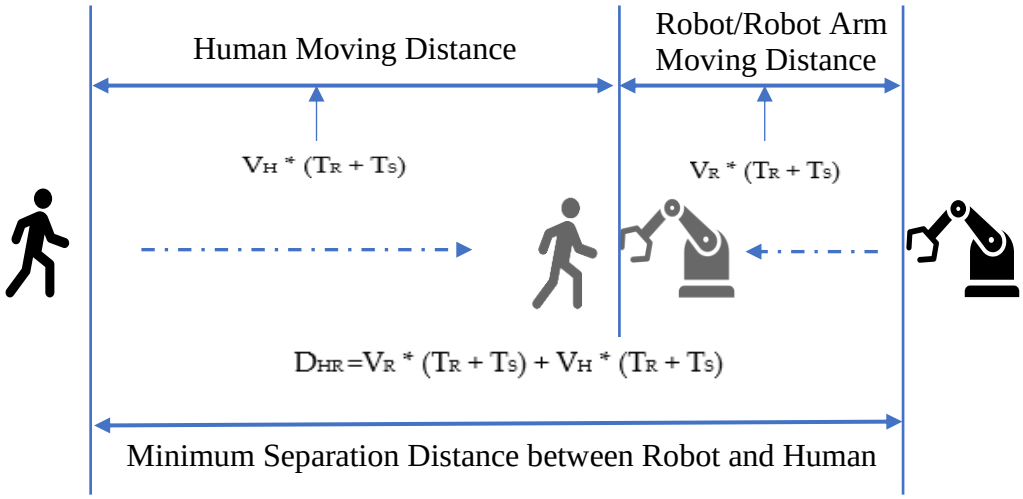


Figure 29. Minimum separation distance

6.3.4.2 Workspace

To align the modelling with a realistic robotic cell, the author introduced the workspace notation. This notation is similar to the BPMN pool/lane (see Figure 30). In BPMN, pools represent companies, departments, or roles. Lanes represent sub-entities within these organizations and appear as swim lanes inside the pool [45]. For robotic processes, there are usually only limited humans involved in the process. In the robotic cell, the focus is not different entities, such as roles or departments, but different areas in the cell. Therefore, the use of pools/lanes as workspaces for separating the different areas in the robotic cell is proposed.

One can use workspaces when tasks involved in the process are carried out in multiple workspaces. A workspace can have input and output and precondition and postcondition assigned to it. With these properties, one can set specific requirements for the workspace. An example of the use of workspace notation is shown in Figure 31.

Figure 30. Workspace notation

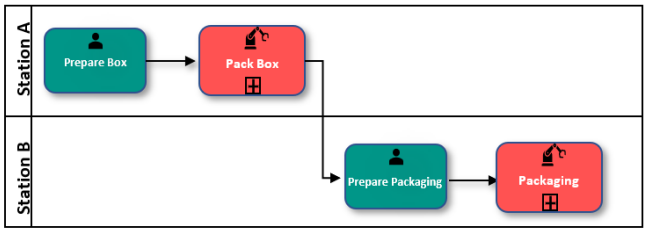


Figure 31. Example of using workspace notation

When tasks switch from one workspace to another, notifications might be needed to inform the robot/human from another workspace to react. For example, there can be a pop-up window or a voice reminder of the workspace change.

6.3.4.3 Decision Making

I modelled decision making with two different notations. The author called one "condition" (see Figure 32), which represents rule-based decision making, and the other "decision" (see Figure 33), which refers to human decision making.

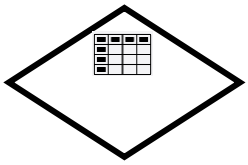


Figure 32. Condition notation

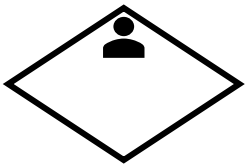


Figure 33. Decision notation

Rule-based decision making is enabled by a rule table where users can specify their decision logic. Table 7 illustrates how the rule table looks, with some examples to demonstrate its usage.

Table 7. Rule Table

Sequence	Parameter	Equation	Value	Action
1	Bin status	=	Empty	Fill up bin
		≠	Empty	Do nothing
2	Distance, robot and human	=	$V_R * (T_R + T_S) + V_H * (T_R + T_S)$	Stop robot
		>	$V_R * (T_R + T_S) + V_H * (T_R + T_S)$	Robots run at collaborative speed
3	Quality check	=	Accepted	Move product to next station
		=	Rejected	Move to rejected bin

The rule table checks whether the conditions are met. Depending on whether the conditions are met or not, a corresponding action is followed. Often, there is only one parameter that is being checked, for example, “quality check”. However, there are also cases where multiple parameters are checked. In this case, the sequence of the condition check needs to be defined. Using a sequence column, the parameters are checked sequentially one by one. When the rule table is defined, this sequential execution must be followed.

Human decision making allows operators/users to specify a question that will be answered at the run time. In this step, a question will pop up and ask the operator/user to give an input (commonly a yes or no answer). Based on what the operator chooses (yes or no), a different branch of the process flow will be executed. Figure 34 gives an example of the decision notation.

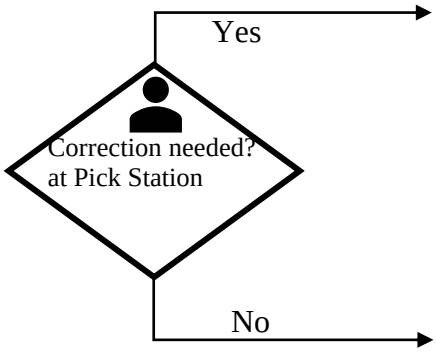


Figure 34. Example of using decision notation

6.3.5 Other Extensions and Modifications

RTMN 2.0 not only extends the HRC area but also covers other additional functionalities. Requirements and KPIs have been added to the modelling language. The aim is to achieve traceability from requirements through tasks, skills, and primitives to reach the defined KPIs. The process notation is added to represent the different processes. The author also made some modifications to the skill and primitive notations in the first version of RTMN so that the distinction would be more obvious.

6.3.5.1 Requirements and KPI

Key performance indicators (KPIs) form a specialized set of metrics steering management activities towards improving company performance, representing quantitative measures of critical success factors (CSFs) in enterprises [123]. They serve as variables reflecting progress toward goals while aligning with the company's vision and strategies [123]. Displaying KPIs is essential in order for supervisors and employees to seek improvements and reach for better performance [123]. In an HRC environment, safety and ergonomics are widely acknowledged as crucial pillars, while in a business context, productivity, encompassing both effectiveness and efficiency, and economics must be considered [123].

Therefore, after analyzing numerous research papers, Caiazzo et al. divided the various KPIs related to HRC into the following four areas: "Productivity", "Economics", "Safety", and "Ergonomics" [123]. The author adopted their systematic approach and categorized the requirements and KPIs into these four categories.

The requirement notation (see Figure 35) and KPI notation (see Figure 36) were added to RTMN 2.0 to visualize the business requirements and KPIs at the process level. Requirements and KPIs at the task level can be assigned if needed.

The requirement notation represents the business requirements of a robotic process/task. It has the following properties: name, formula, KPI, input, output, precondition, and postcondition. One can set constraints using preconditions and postconditions. For example, one can add a postcondition—the sum of net machine utilization and net operator actuation is 100%. The KPI notation defines the key performance indicators linked to the requirements (see Figure 38). Table 8 illustrates the general KPIs for the HRC robotic processes, among

which a few are related to humans: accident rate, net operator actuation, and human exposure to chemicals.

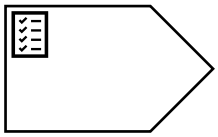


Figure 35. Requirement notation

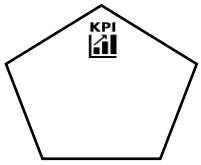


Figure 36. KPI notation

Table 8. Requirements and KPIs

Requirement	Requirement Formula	Name of KPI	KPI Formula
Decrease cycle time by 10%	(Cycle Time old - Cycle time new)/Cycle Time old ≤ 10%	Cycle Time	Cycle Time = Process end time - Process begin time
Increase productivity by 20%	(Productivity old - Productivity new)/Productivity old ≥ 20%	Productivity	Units of product/ production time (hours)
Reach success rate of 80%	Success Rate ≥ 80%	Success Rate	Nr of success action/ Nr of total actions
Reach reprogramming time of less than 10 minutes	Reprogramming Time < 10 min	Reprogramming Time	Time finish reconfiguration - Time start reconfiguration
Reduce scrap products to less than 5%	Nr of scrap product/Nr of total product ≤ 5%	Scrap Product	The number of products not accepted in quality inspection
Increase machinery utilization to more than 75%	Net Machine Utilization ≥ 75%	Net Machine Utilization	Machine run hours per process/process duration
Reduce net operator actuation to less than 25%	Net Operator Actuation ≤ 25%	Net Operator Actuation	Human working hours per process/process duration

Reduce accident rate to 0	Accident Rate = 0	Accident Rate	The number of reportable health and safety incidents per month
Increase human safety /reduce exposure to chemicals/danger	1 - Human exposure to chemical rate ≥ 30%	Human Exposure to Chemicals	Time of human exposure to chemicals/danger/cycle time

6.3.5.2 Traceability of Requirements and KPIs

The inputs for calculating the KPIs come from ROS, the robot control level. From primitives and skills, the input values can be obtained, can be used them for calculating the KPIs, and can validate the requirements. Through this, traceability of the requirements and KPIs is achieved.

To be more specific, KPIs are calculated automatically based on the values created in different layers of the robot system. For example, for a "pick" skill, it is tracked whether the pick is successful or not. The success rate of picking is calculated by the number of success picks/total pick attempts. Another example is the robotic task scrap product, through which quality inspections of the number of accepted products and unaccepted products are tracked. If the product is within the error tolerance, it will be accepted, and vice versa. The system can add up the number of accepted products and the number of unaccepted products, and then product quality is calculated by dividing the number of products not accepted by the number of total products.

6.3.5.3 Robotic Process

A robotic process involves robots in the process. The author has added this notation (see Figure 37) because it shows the relationship between business requirements and the overall process visually. A process can contain human tasks, robot tasks, and HRC tasks.

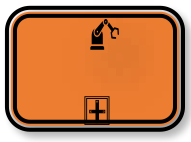


Figure 37.Robotic process notation

The robotic process has data properties: name, owner, and organization. Requirement/Objective and KPIs are no longer just properties of the process. They have their own notations, as described earlier.

An example of modelling KPIs and requirements for a process is shown in Figure 38. The dashed arrows indicate data association.

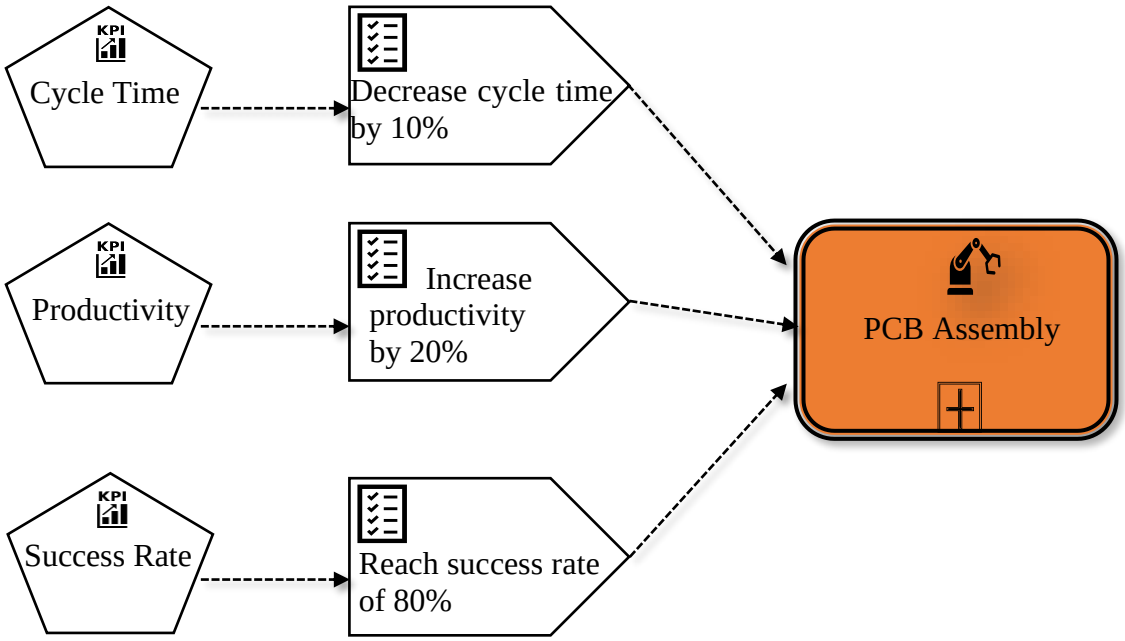


Figure 38. Process-requirement-KPI example

6.3.5.4 Skill and Primitive

I modified the icons of the skill and primitive notation in order to easily distinguish the robot task and robot skill visually whilst bringing the appearance of the skill and primitive notation closer together. Then, both skill and primitive had robot body icons (see Figures 39 and 40). The robot task has a robot arm icon.

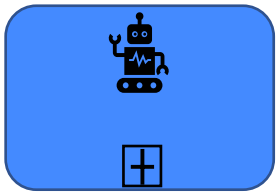


Figure 39. Skill notation

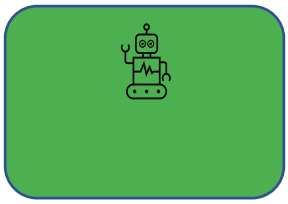


Figure 40. Primitive notation

6.3.6 The Implementation

As described in the former RTMN paper [72], the modelling language RTMN was used in the graphical user interface (GUI) of the ACROBA platform. The notations from RTMN 2.0 were defined as drag and drop elements in the GUI. The GUI was developed according to the Flutter framework [124]. Flutter uses Dart as its programming language. The special package Dartros [51] enables Flutter to communicate with ROS [16] by implementing the ROS client library.

The GUI using RTMN 2.0 serves as a modelling tool. The users can create a process model in the GUI. The process sequence is then sent from the GUI to the task planner through Dartros [125]. The author also described the task planner in another research paper [126]. The task planner takes the sequence and "translates" it to a behavior tree, which is connected to ROS. The task planner also provides feedback and the status of the tasks to the GUI. The amount of available modelling elements, such as skills and primitives, is managed through an ontology where domain knowledge such as tasks, skills, and primitives are defined. The characteristics of ontology ease the sharing of knowledge, provide clearly structured data, and enable accurate reasoning of the data [127]. Therefore, the benefits of ontology for knowledge sharing among heterogeneous systems and human–machine interactions were also adopted.

6.3.7 Demonstration

The design of the Graphical User Interface (GUI) for RTMN 2.0 is presented in Figure 41. The left side is divided into different abstraction levels: processes, tasks, skills, and primitives. At the bottom left conner, the connecting notations are shown. One can simply drag and drop the notations on the left to model an HRC process.

I illustrated one example of an HRC process. This example is a modification of a use case process due to confidentiality. A Printed Circuit Board (PCB) assembly is a process used to assemble a PCB. The assembly works by putting different kinds of Pin through Hole (PTH) components onto the PCB. This process consists of three tasks: moving PTH to pick station, assembling PCB, and inspecting PCB. Move PTH to pick station and assemble PCB are two HRC tasks, while inspect PCB is a robot task.

For the HRC task of moving PTH to pick station, safety mode SMS should be ensured because it is a sequential HRC task with the human and robot working one after another. It has a human task, fill up bin; followed by a robot skill, move to bin; then a primitive grasp PTH; a robot skill, move to pick station; and a primitive, release PTH.

For the HRC task assembling PCB, safety mode SSM should be ensured, as humans and robots work in parallel for this task. They work side by side on the same PCB, but not at the same time. When a PTH is wrongly positioned, the human needs to correct it. As the human approaches the robot and enters the "red" zone, the robot stops. This task consists of a simple robot skill—match PCB (this skill scans PCB, then, based on the scanned image, loads the CAD model of the PCB)—and two parallel tasks (combination of skills)—correct position error by human and pick and place by the robot. The join/split notation is used here to split the sequence into two then merge the two back into one again.

The robot task inspecting PCB does not involve any humans. It is purely performed by a robot. For this task, a condition notation is used to decide where the PCB goes after inspection. If it is accepted, it is moved to the next station. If it is rejected, the PCB is scrapped.

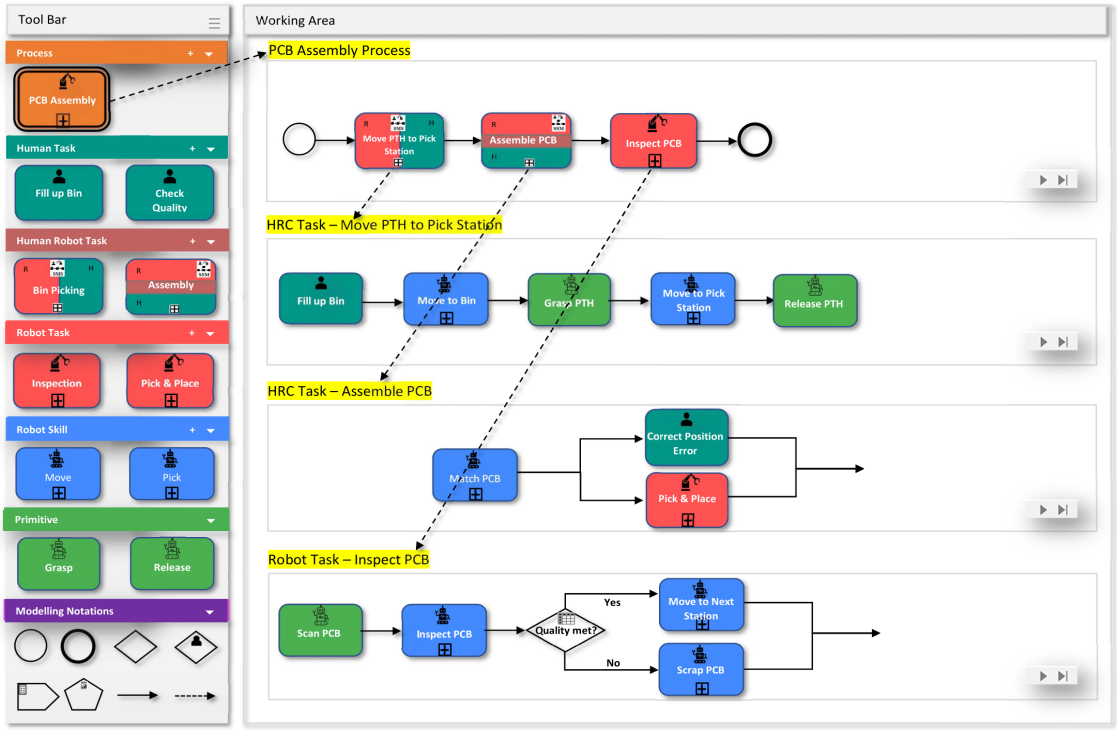


Figure 41. HRC process example—PCB assembly

6.3.8 Validation

The first round of validation was carried out by interviews and questionnaires. These were described in Chapter 5. Another round of evaluation of the GUI is planned after its implementation. Both quantitative and qualitative research methods will be applied. The author will analyze the results of the questionnaire and interviews and assess the benefits and drawbacks of the GUI. Based on the feedback, further improvements will be made.

There is a lengthy validation process after rollout to the five real-world industrial use cases. This will validate cost saving, productivity improvements, safety assurance, and efficiency. The form that will be used to collect data is presented in Appendix 15.6.

There are two demonstrations for the use of the GUI and Human Robot Collaboration functionality added in RTMN 2.0 in Chapter 9 Section 9.2 Scenario 1: Robot Programming through the GUI Using RTMN presented the use of RTMN in a robotic cell. Section 9.4 Scenario 3: Human Robot Collaboration shows how a collaborative Human Robot Collaboration task is modelled and how a human worker and a robot working together in parallel in the same robotic cell.

6.3.9 Discussion

This chapter has not yet covered the details of modelling of human factors related to mental safety (mental stress and anxiety) induced by close interaction with robots. According to Hancock et al., establishing trust and acceptance is crucial in automation [130]. Norman emphasized user-centered design and its pivotal role [131]. He explained that poorly designed interfaces contribute to stress. Wickens et al., in their human factors engineering research, found out that effective workload management is essential to prevent cognitive and physical overload [132]. Asaro worked on the ethical and psychological impact of interacting closely with robots, and he described that these closed interactions may raise ethical concerns and have psychological implications, such as fear or discomfort [133].

Although challenges related to mental safety in HRC have been discovered, there are still many difficulties when addressing these changes: 1. collaboration and coordination between experts across various areas, such as robotics, psychology, human–computer interaction, and ethics; 2. the difficulty of keeping up with rapid technological advancements; 3. different people act differently when working with robots; and 4. The author does not have a complete understanding of human cognition and emotion.

The author will continue their research in this area and aim to develop a comprehensive approach for modelling human factors. The plan is to address the following factors in future research: performance factors, such as productivity, capability, intension, and workload, and health factors, such as fatigue, anxiety, satisfaction, trust, and acceptance. Methods for assessment of these factors will be defined using system knowledge and facts saved in an ontology. This assessment can be defined in the form of rules for the HRC tasks. Based on the conditions of the rules, actions will be triggered for task planning and execution.

6.3.10 Conclusions and Outlook

This chapter covers the extension and modification of the RTMN. RTMN 2.0 focuses on the modelling of human robot collaboration processes. Modelling of the requirements and KPIs was added to enable traceability from a process to the lower robotic control level. Notations related to decision making are modified as an improvement to RTMN. These additional features of RTMN 2.0 together extend RTMN to the HRC process domain by separating the HRC tasks from normal tasks (human tasks and robot tasks) and labelling the different types of human–robot interactions. The five separation notations are self-explanatory, and users can use them as predefined templates, to easily design the task details, and to ensure safety through safety modes, which require configuration for hardware and software implementations. The safety of humans is ensured through the assigned safety mode of each notation, as each safety mode is monitored in the background with corresponding rules to control safety. These additional features provide the following benefits to users: increase intuitiveness, provide easy-to-use templates, is code-free, and ensures safety and reusability. In addition, the basic RTMN notations are very similar to the notations users use to draw process flows. The familiarity then eases the utilization of the tool, therefore allowing people without programming expertise to plan and control robots in an easier way.

User training is planned before the final testing after the implementation. This training is expected to be within 10 hours. The simplicity of the tool is very high in comparison to gaining high-level robot programming skills, which would take months or years.

7 ONTOLOGY SUPPORT ROBOT PROCESS MODELLING

7.1 INTRODUCTION

Robots and robotic manipulators play a role of fundamental importance in the manufacturing industry, particularly for complex but repetitive tasks. However, the success of industrial robotics to date has depended strictly on a controlled, static environment [32]. Therefore, the industry requires advancements in robotic systems to reduce the programming effort and to adapt to changing environments due to the new requirements of agile production. The field of cognitive robotics deals with the development of robot systems equipped with awareness of their capacities and limitations in changing environments and agile production [134].

Ontology-based knowledge representation has been studied for its functionality in achieving robust autonomous robotic systems. When such a system is equipped with knowledge representation, the cognitive skills provided to the robot allow it to perform tasks, reason the knowledge base, and interact with various environments autonomously [6]. Ontology methods provide formalized concepts, which may then be used to construct a knowledge base using classes, axioms, and relationships. Thus, a shared domain vocabulary may be defined, making both collaboration and reusability simpler tasks [34]. Characteristics of ontology include formalization and standardization. Ontology provides clear structure and accurate data through which knowledge sharing and reusing among heterogeneous systems is made more convenient [9].

This research is supported by the European Horizon 2020 ACROBA project [1]. This project aims to develop an AI-driven cognitive robot platform that can be easily and flexibly adapted to different agile manufacturing scenarios. There are five industrial pilot lines in this project. The research must support the realization of the five industrial pilot lines. Therefore, these industrial manufacturing processes provide not only requirements and inputs for the research work but also targets for the validation process.

IEEE has developed the standard ontologies for robotics and automation [135]. This standard consists of a Core Ontology for Robotics and Automation (CORA) and other ontologies to support CORA. CORA covers the fundamental concepts, their relations, and axioms in this domain. CORA is aligned with the Suggested Upper Merged Ontology (SUMO), which is an upper-level ontology that provides definitions of basic ontological concepts for all domains to use as a foundation [136]. To enable ontology reuse, it is recommended to build upon existing ontologies and standards. However, the detailed concepts regarding robotic processes are not defined in this ontology.

Through the literature review, it was found that although the use of ontology for task-planning in robotics applications has been researched, its implementation in industry has yet to see major advancements. In the literature, the use of terms and concepts in robot process area are not strictly regulated, and there is a need to provide a standardized use of terms to avoid misuse, misunderstanding, and miscommunicating of concepts in this area. KPIs are used to evaluate fulfillment of requirements. They are usually defined at the process level. However, in the robotics area, “task” is often the central focus instead of “process”. If measurements and controls are designed on the robotic task level while KPIs for requirements are set at the process level, the traceability of the requirements cannot be easily tracked to the robot control level. Therefore, the link between process to task needs to be defined.

For the aforementioned reasons, the author has developed a more specific domain ontology based on CORA to the robotic process domain. This ontology provides additional knowledge for the "process" and "task planning" concepts. It defines the terms and their relationships in this domain as well as provides the link from KPIs (high-level process management layer) to robot actions (low-level robot control layer). With this link, the KPIs defined for the process can be calculated automatically by the robotic system based on robot actions. Pedersen et al. have presented a robot skill model that is well accepted in the robotics industry [137]. It contains three abstract layers: task, skill, and primitive. This model has helped in determining the terms for domain ontology.

In the following sections, a problem description is presented first with a systematic literature review of the state of the art, and gaps in the current research are identified. In the research method section, the ontology development process is described, and a new ontology is proposed. Then, there is prototyping and the applications of the ontologies in the ACROBA project [1] are described. Finally, conclusions are drawn and an outlook for future research is provided.

7.2 Literature Review and Problem Description

7.2.1 Literature Review

Planning complex robotic processes in a real-world environment requires sophisticated domain knowledge, which can be acquired from ontology-based knowledge representation and reasoning techniques. Using knowledge representation provides cognitive skills, which enable robotic platforms to perform tasks, make decisions, and interact with various environments [6]. The benefits of implementing ontologies for robotic task planning are investigated, and to identify any gaps in this field, two research questions are raised:

- Is there an existing ontology for robotic process planning that covers the whole industrial manufacturing domain - light out manufacturing and collaborative manufacturing?
- Is there any traceability of high-level process KPIs to low-level robotic control?

Research related to ontology-based robotics task planning covers different robotic application areas such as human-robot collaboration, autonomous robotics, and industrial manufacturing. There are two papers [5] and [138] that thoroughly reviewed the ontologies in the robotics domain. They provided us with good foundations and contributed a lot to our literature review. Each area will be touched on in the following paragraphs.

7.2.1.1 Human-robot Collaboration

A large amount of research has been conducted on the use of Ontologies for Human– Robot Collaboration (OCRA) [139]. OCRA is an Ontology for Collaborative Robotics and Adaptation. This ontology focuses on collaboration and plan adaptation. When executing industrial robotic tasks, the robot collaborates with humans and is able to decide what to do based on the adaptation plans with the support of the ontology [9]. However, the validation of OCRA is limited to one scenario of a human and a robot collaboratively filling a tray [5].

Sharework Ontology for Human–Robot Collaboration (SOHO) describes an ontology for Human–Robot Collaboration that aims to flexibly represent the collaborative production process [7]. The main drawback is that SOHO is not developed for realistic collaborative industrial scenarios. Umbrico et al. [140] propose an extension of SOHO. An experimental evaluation was carried out on the developed representation and reasoning technology, justifying its efficacy. However, this work focused solely on Human–Robot Collaboration scenarios and has yet to be developed further. A knowledge storage and reasoning framework, Ontologenius [141] allows to modify the knowledge base during execution, and it maintains these changes even when powered off. This system has been tested in human-related scenarios.

7.2.1.2 Autonomous Robotics

Extensive research work for ontologies can also be found in autonomous robotics. ORO provides a framework for knowledge storage and reasoning. In this framework, robots are enabled with cognitive skills to perform different tasks such as task planning and replanning and collaboration with robots and humans in a complex environment [5]. The Autonomous Robot Architecture Ontology (ROA) [142] aims to provide a conceptual framework allowing people and robots to share and exchange information about robot architecture. The conceptualization is similar to the meta-models of concept representation languages such as the Unified Modelling Language (UML) [44]. Yuguchi et al. [143] explores the usage of ontologies for home service robots to equip them with the ability to perform tasks using home appliances. However, the framework could only execute sequential tasks to control home appliances. Robot Task Planning Ontology (RTPO) is an indoor service-oriented robot task planning ontology [137]. Based on this ontology, the author developed a task planning algorithm which is used for robots to autonomously assign high level tasks [137]. Beetz et al. introduced KnowRob2 [8], based on its predecessor, KnowRob [144] – which is an advanced knowledge processing system that accomplished the performance of a multitude of complicated manipulation tasks such as making a pizza, conducting chemical experiments, and setting tables. The aim of KnowRob2 was to improve the capabilities of robotic agents to acquire open-ended manipulation skills and reasoning when faced with realistic manipulation actions. KnowRob2 was proposed as a query- answering system, where queries are connected to tasks and refer to images captured by the agent and its motions. Although KnowRob2 is advanced, it has not yet been linked to any industrial scenarios.

7.2.1.3 Industrial Manufacturing

Manufacturing does not have many ontology-based task planning-related research. Weser et al. presented an ontology-based metamodel for manufacturing capabilities for production resources. Their paper describes a hierarchical ontology architecture that formally defines capabilities as abilities of manufacturing resources for different domains and use cases. However, this paper did not touch on the abstraction of robot skills [145]. In the German Basys 4.0 Initiative, Perzylo et al. also pointed out that capabilities are important and formally modelling them enables the interoperability of hardware and software. In this paper, the concept of using manufacturing skills is detailed [146]. The Factory of the Future ontology (FoF) [147] attempted to model assembly tasks needed for product assembly as well as the robotic skills

needed to perform these tasks. Although the results of this project indicated that the use of ontology is practical in establishing a common understanding between all involved components, the FoF ontology is limited in its ability to model the current state [147]. The author pointed out that in the future, they aim to extend FoF for HRC and more manufacturing methods. ROSETTA [145] attempts to characterize capabilities of manufacturing resources using a formal definition of capabilities. It implemented a set of ontologies of robot skills aiming to create intelligent support for reconfiguration and adaptation of manufacturing robotic cells. This paper indicated that the formal representation of ontologies is suitable for encoding knowledge, and ontology models are easily maintained and reused. Saxena et al. [148] present a knowledge engine RoboBrain, which may learn and share knowledge representations such that robots may carry out various tasks. However, RoboBrain is still in the experimental stage.

7.2.2 Problem Definition

The findings are summarized as follows:

- There is no existing ontology for robotic process planning that covers the entire manufacturing process. Most of the ontologies are focused on the service robotics domain aiming to achieve robot autonomy.
- There is an increasing amount of effort that has been put into Human–Robot Collaboration (HRC). However, in the industrial robotics area, the amount of work is less extensive.
- No key performance indicators (KPI) were included for robotic process planning and control because the research focus was mainly on the robot control level, which is the closest to the robot. Process on the other hand is at the top. It goes down to task, and then the task can be performed by robots.
- The research focus is either tool-centric or product-centric.

The literature review indicates clearly that while several attempts at developing ontology-based knowledge frameworks for robotic systems have been made, none of these have been developed to a state advanced enough to be adopted for and used to fully operate industrial processes. The use of ontology has been explored extensively for situations involving human–robot interactions. This is mainly because a level of ambiguity exists between human instructions and what a machine understands. While some papers did focus on ontology-based task planning for robotics in industrial systems, these projects were experimental and required further work. Therefore, it can be concluded that ontology-based task planning for robotic industrial systems is a field that requires further work to develop a robust system, which could be adopted in industry.

The author notes that there is hardly any work that describes the robot process level. Process level refers to the level where objectives are set and KPIs are measured for performance evaluation. It is different from the task level where the focus is on robot actions. This fact creates a big gap between performance management and robot control. At the process level, KPIs like cycle time, accident rate, product quality, etc. need to be calculated and measured for management reporting. When process management is separated from robot control, it is hard to evaluate the performance of the processes, and it is especially difficult to know how to reach a better KPI level by optimizing the lower-level robot performance.

The research in this field has two focuses: tool-centric (a manufacturing task is de- scribed based on available hardware and software components) and product-centric (de- scribing the

product and associated production steps independent from specific production resources). To profit from both approaches, not only do robot skills need to be implemented but also the capability of the production resources needs to be modelled [145].

7.3 METHODS - ONTOLOGY DEVELOPMENT

In the past years, researchers have proposed various kinds of ontology development approaches. Here a combined approach that considers both the SABiO ontology development approach [149] and the hybrid framework used by [150] was followed. The overview is shown in Figure 42.

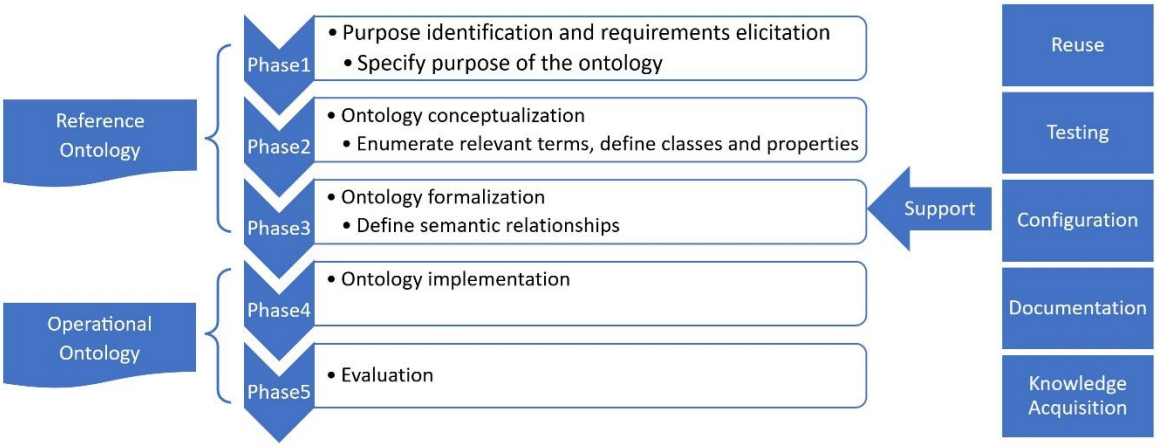


Figure 42. Combined Ontology Development Methodology (Created based on [149] and [150]).

In this merged approach, there are five main phases that represent the main development process. The first three phases are used to develop a reference ontology, and the last two phases are used to develop an operational ontology. Reuse, testing, configuration, documentation, and knowledge acquisition describe the supporting process in ontology development. The SABiO method differentiates the reference ontology and operational ontology and divides the ontology development processes into main and supporting processes [149]. It provided guidance for these processes. The hybrid framework [150] has very similar phases to the SABiO method, but it provides more detailed structures for the different phases. For instance, in Phase 2, it recommends enumerating relevant terms, defining classes and their properties. The reason to combine these two is to profit from the advantages of both methods.

The reference ontology and operational ontology are described in SABiO as follows:

"A reference ontology is a special kind of conceptual model. An operational ontology,

in turn, is a machine-readable implementation version of the ontology, designed with the focus on guaranteeing some desirable computational properties" [149].

Therefore, the first three phases are used to develop a reference ontology, and the last two phases are used together with the first three to develop operational ontology. The details of the reference ontology development are described in the following sections. Section 4.3.1 focuses on identifying requirements for ontology, ontology conceptualization, and ontology formalization.

7.3.1 Ontology Purpose Identification and Requirements Elicitation (Phase1)

The goal of this step is to identify the purpose of the ontology and its intended uses. This step is crucial as it generates the requirements that are the prerequisite for constructing an ontology. This step involves ontology engineers, domain experts, and potential ontology users. Figure 43 shows the overview of this phase, and it is taken from [149].

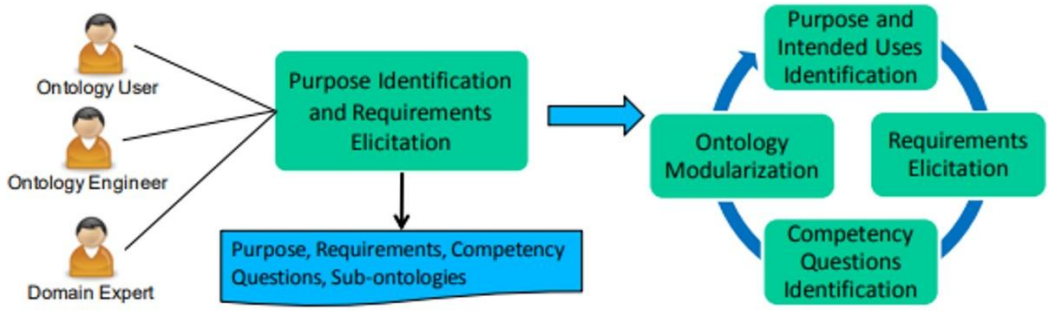


Figure 43. Ontology Purpose Identification and Requirements Elicitation (Source: [149]).

7.3.1.1 Competency Questions

To determine the scope of the ontology, we need to find out what is relevant and what is not by using Competency Questions (CQs). These CQs capture the functional requirements for the ontology. We use the top-down strategy for forming CQs. The complex questions are asked first and then they are decomposed into simpler ones. In Table 9, a set of competency questions are listed.

Table 9. Competency Questions and Answers

CQ Nr	Competency Question	Answer
CQ1	What is the purpose of the ontology?	The ontology aims to support and facilitate the robotic process automation and human robot collaboration
CQ2	What is the scope?	The ontology will include information on robotic processes, this includes pick and place, inspection, assembly, robot control, task planning, etc.
CQ3	Who are the intended end users?	There are two groups of intended users: 1. system engineers, programmers, software engineers for support in AI functionalities. 2. Operators and engineers for error diagnostics and system queries.
CQ4	What is the intended use?	To support robotic process modelling To support PDDL robotic task planning To create the sematic mapping from process goal to robot actions. To support human robot collaboration To create a common understanding of the terms in the domain between hardware and software engineers, programmers, operators, process designers so that knowledge is communicated consistently.
CQ5	What types of robotic processes for manufacturing exist?	There are two general types, one is applying robotics in the light-out manufacturing processes. In these processes the main goal of robots is replacing human workers and achieving automation. The other type is to apply robotics in collaborative manufacturing processes. The main goal for these processes is for robots and humans to work together safely to increase efficiency and/or improve human work conditions.

CQ6	How do robotic processes work?	These are processes that involve robots in the manufacturing steps. The process must enable communication with robots and robot control.
CQ7	What are the robotic processes composed of?	A robotic process consists of different tasks, it can be human tasks, robot tasks or combined tasks that are performed by human and robot together
CQ8	What are the links between these components?	These tasks can be independent or dependent. Sequential tasks are performed one after another (often the output of the former task is the input of the later task). Parallel tasks can be performed at the same time without influencing one another.
CQ9	What impacts the robotic processes?	The speed of the robot, the performance of the robot, the safety of the robot, the human factors (such as trust, fatigue, stress), the error diagnostic
CQ10	How are robotic processes evaluated?	There are KPIs that measure the process performance, such as cycle time, productivity, accident rate etc.
CQ11	What are the types of robotic tasks?	human tasks, robot tasks, human robot collaboration tasks that are performed by human and robot together.
CQ12	What are robotic tasks consisting of?	Tasks can contain subtasks. Robot tasks often consist of many robot actions that are called by different names in the research are and in practice, for instance skills, capabilities, functions etc.
CQ13	What are the links between these components?	It is similar to tasks; these components can also work dependently or independently.
CQ14	What are types of human robot collaboration tasks?	They are different based on the task types (coexistence, sequential, parallel, cooperate, teaching, collaborate) and the collaboration mode (Fence, safety-rated monitored stop, hand guide, speed and separation monitoring, power and force limiting).
CQ15	What impacts human robot collaboration?	Human factors (how humans feel and behave around the robot), safety of the robot, safety measures, safety control.
CQ16	How do humans and robots work together?	Humans and robots can work together in different ways. :coexistence (working on their own, do not have anything to do with each other), sequential (they do the work sequentially one after another, they are dependent), parallel (they work at the same time, but not on the same part), cooperate(they work on the same part, but not at the same time), teaching (human guide the robot to learn something, human is in control, robot in teaching mode), collaborate (human and robot work together on the same part and at the same time)
CQ17	How are safety ensured when humans and robots work together?	There are different safety modes based on ISO standards, these safety modes must be ensured to guarantee safety.

CQ1–CQ4 are top level questions that cover the ontological rationale. CQ5–CQ17 are more detailed questions used to elicit domain requirements.

7.3.1.2 Modularization

As the domain knowledge is quite complex, the target ontology is broken down into sub-ontologies: core robot process ontology and application specific ontologies. The core robot process ontology contains the generic robot process ontology, and it is called ORPP. The ACROBA project-related concepts are planned in the application ontology named ACROBA.

This ACROBA ontology will gather project-specific and use case-specific knowledge. The concepts related to the graphical user interface (a tool to model robotic processes) are planned in an application ontology called RTMN. RTMN is the name of the modelling language [151]. RTMN has different notations such as Process, Task, Skill, Primitive, Sequence Flow, Condition, Decision, Error, Start/End, etc. The details are defined in the RTMN paper [151] and its successor RTMN2.0 [152]. With the support of this core robot process ontology, the graphical user interface (GUI) generates its components (ontology helps populate the GUI). The concepts related to task planning are in the PDDL application ontology. PDDL stands for Planning Domain Definition Language [153]. It is a planning language widely used in the robotic industry to define planning problems. Here the focus is on the core ORPP ontology.

This ontology aims to contribute to the robotics manufacturing domain and opt for a standardization that simplifies robotic task programming. The characteristics of the ontology are described as follows:

- achieves modularity, reusability, extensibility, knowledge sharing, and knowledge reasoning
- contains process, task, and robot action knowledge
- creates a link between processes management and robot action
- based on upper-level ontology, extends domain-level ontology, and adds application-level ontology
- combines the tool-centric approach (a manufacturing task is described based on available hardware and software components) and the product-centric approach (describing the product and associated production steps independent from specific production resources) [145]
- supports the populating of a graphical user interface (GUI) to design, create, simulate, and run the robotic processes
- supports the PDDL task planning

7.3.2 Ontology Conceptualization and Formalization (Phase2&3)

7.3.2.1 Conceptualization

The ontology conceptualization phase requires knowledge acquisition. Based on [150], the process of extracting knowledge is divided into three steps. Firstly, find the relevant terms for the ontology. Secondly, define classes based on the terms. Thirdly, specify properties for the classes. Figure 44 gives an overview of two phases, and it is taken from [149].

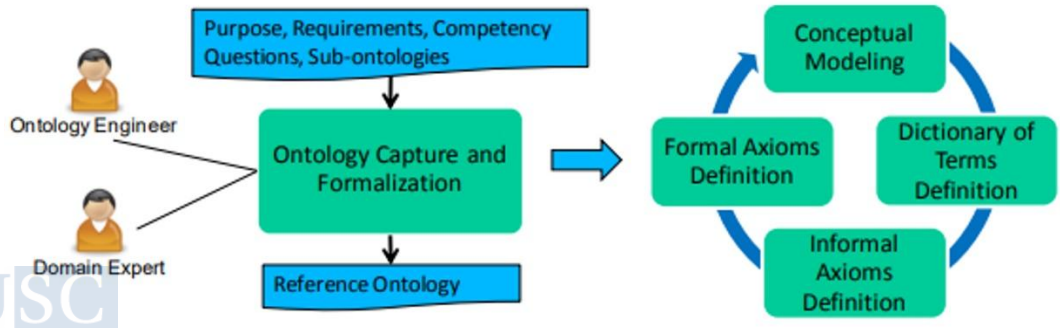


Figure 44. Ontology Conceptualization and Formalization (Source: [26])

7.3.2.1.1 Find the relevant terms for the ontology.

For ontology-based systems to be effective, they must involve standardized sets of terms that define a common understanding of the domain and may be shared amongst different systems [146], [154]. To do that the most used terms in this area must be identified.

1) From Literature

A literature review was conducted to capture the knowledge of robotic process ontology. In Figure 45, the commonly used terms are listed. For each reference, the year of the publication, the name of the ontologies, their application fields, ontology language used to model the ontology as well as the relevant terms they used in their ontologies are presented in Figure 45. The "y" is an abbreviation for "yes", which indicates that this term is mentioned in the ontology. The total number of the terms noted in the different literature is calculated. Among the most used terms in the different papers, "Task", "Skill", and "Primitive" are identified as the most common terms. Sometimes nomenclature is not uniform (e.g., functions, capabilities), but the meanings fall into these main categories—task, skill, and primitive. Hardly any ontology addresses the process level. KPIs are also missing from the existing ontologies. Therefore, there is a need to standardize the terms used and shared in the robotic process domain. This will provide a common understanding of the domain and allow the ontology to be shared amongst different systems. These reusable knowledge pieces can be used to ease robot programming. To agree on the meaning of terms, the definitions in literature are checked first.

Year	Name	Application Field	Ontology Language	Terms												
				Task	Skill	Primitive	Process	Action	Behavior	Plan	Goal	Capability	Funtion	Device	Collaboration	Agent
2009	KNOWROB	service robot	Web OWL	y				y	y			y	y	y		y
2010	ORO	collaborative robot	RDF triples													
2014	RoboBrain	service robot	Graph structure													
2017	ROA	service robot	OWL	y	y	y	y		y		y		y			
2018	KnowRob2.0	service robot	Web OWL	y				y								
2018	ROSETTA	inductial robots	OWL	y	y	y					y			y		
2019	RTPO	service robot	OWL	y				y								
2019	BaSys 4.0	inductial robots	OWL	y	y	y						y		y		
2020	SOHO	collaborative robot	FOL	y	y	y						y	y			y
2020	C4I	industrial robot&collaborative robot	OWL	y	y	y						y				
2021	FoF	service robot	OWL	y	y											
2022	SOHO extension	collaborative robot	OWL	y		y						y	y		y	y
2022	OCRA	collaborative robot	FOL,OWL DL							y	y				y	y
2022	OTP	service robot		y							y					
2022	Rosetta extension	inductial robots	OWL	y	y	y										
			Total	12	7	7	1	3	2	1	4	5	4	2	2	4

Figure 45. Analysis of Ontology Terms Related to Robotic Task Planning.

There are different definitions of the term "Task" in the literature. Many researchers define task as "a sequence of skills with specified parameters and conditions that aims to solve a specific goal [7,137,138,145,147,155]. KnowRob 2.0 uses DUL (DOLCE+DnS Ultralite) action to classify tasks, and it views a task as an even type that classifies an action to be executed [5]. Olszewska et al. consider a task as the interpretation of a robot of what the operator wants to achieve on the robot behavior level [142].

The term "Skill" is also used in different ways in literature. Pederson et al. see robot skills as "intuitive object-centered robot abilities that allow robots to complete a task, which can easily be parameterized by a non-expert" [137]. Some authors describe skills as a

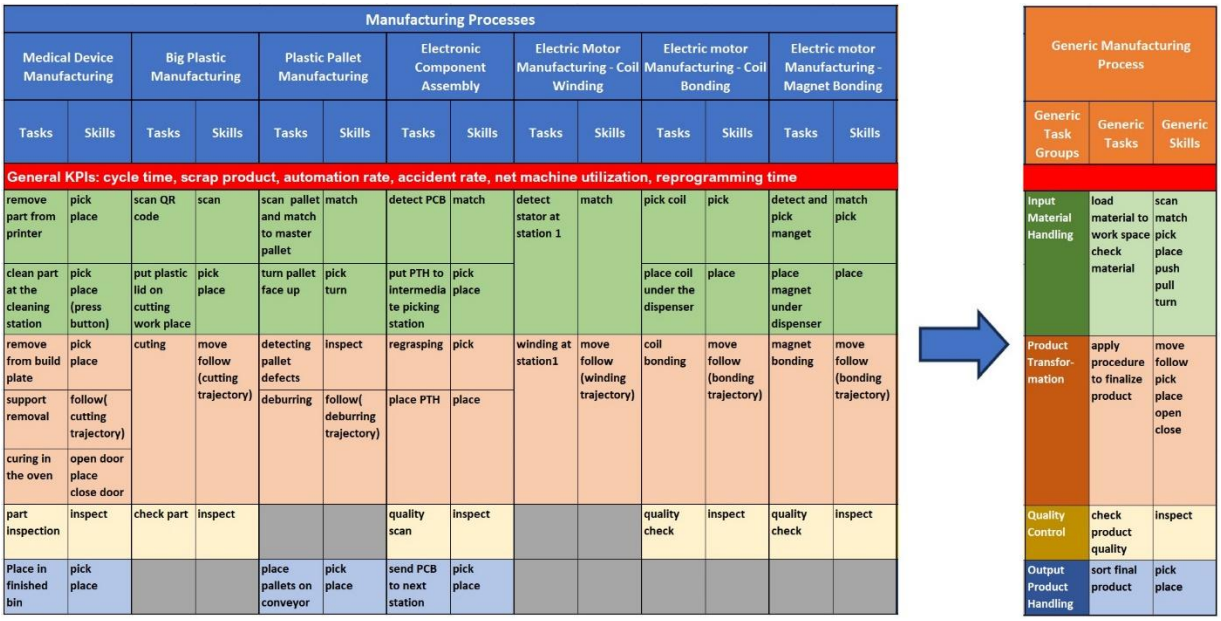
parametrizable and executable functionality to achieve a particular task or goal [145,147]. Others use “abstract capability of a device” [5,138] or “low-level operations/primitives” [7] or “observed/desired actions of the robot” [142] or “robot programs” [156].

The description/definitions of the term “Primitive” vary. Pedersen et al. pointed out that primitives are single operations related to the robot system such as motion planning and gripper opening [137]. Schou et al. defined “device primitives as a set of device functionalities that abstract specific implementation details” [155]. Umbrico et al. focused on capability and defined interaction primitives as ActingCapability and SensingCapability [7]. Olszewska et al. [142] use “function” to describe primitive, which is similar to Pedersen et al.’s definition.

2) From Use Cases in the ACROBA Project

In addition to the literature review, we also analyzed the five use cases in the ACROBA project. In Appendix 15.1 five industrial pilot lines, their goals and challenges are presented. There are seven processes from the five use cases that are in scope of the ACROBA project: medical device 3D printing, big plastic manufacturing, plastic pallet bur handling, electronic components assembly, electric motor manufacturing coil winding, coil bonding, and magnet bonding.

I started the analysis by breaking the processes down into different tasks groups. And then, the different task groups were split into tasks. After this step the tasks were further divided into smaller steps that are called skills. Figure 46 shows the results of the analysis. The columns with blue header cover the analysis of the 7 processes, and the columns with orange header cover the generic definitions of a manufacturing process after consolidating the 7 processes. The generic task groups, generic tasks and generic skills are derived from the consolidation of the 7 processes. The author has identified four generic task groups: input material handling (material preparation for production), product transformation (the core step for producing a product), quality control (product quality check) and output product handling (sorting the products to different places). In each of the task groups the generic tasks are identified. Afterwards generic robot skills are derived from these generic tasks. These skills are match (to recognize and map the scanned material to the master data), pick (to pick up something from somewhere), place (to put something somewhere), move (move from one location to another), push (to move forward), pull (to move backward), turn (to move with an angle), follow (to generate and execute a trajectory), open (to open something), close (to close something), inspect (to scan and check the quality). In the future, the author plans to analyze more production processes to derive more generic skills.



Manufacturing Processes													
Medical Device Manufacturing		Big Plastic Manufacturing		Plastic Pallet Manufacturing		Electronic Component Assembly		Electric Motor Manufacturing - Coil Winding		Electric motor Manufacturing - Coil Bonding		Electric motor Manufacturing - Magnet Bonding	
Tasks	Skills	Tasks	Skills	Tasks	Skills	Tasks	Skills	Tasks	Skills	Tasks	Skills	Tasks	Skills
General KPIs: cycle time, scrap product, automation rate, accident rate, net machine utilization, reprogramming time													
remove part from printer	pick place	scan QR code	scan	scan pallet and match to master pallet	match	detect PCB	match	detect stator at station 1	match	pick coil	pick	detect and pick manget	match pick
clean part at the cleaning station	pick place (press button)	put plastic lid on cutting work place	pick place	turn pallet face up	pick turn	put PTH to intermedia te picking station	pick place			place coil under the dispenser	place	place magnet under dispenser	place
remove from build plate	pick place	cuting	move follow (cutting trajectory)	detecting pallet defects	inspect	regrasping	pick	winding at station1	move follow (winding trajectory)	coil bonding	move follow (bonding trajectory)	magnet bonding	move follow (bonding trajectory)
support removal	follow(cutting trajectory)			deburring	follow(deburring trajectory)	place PTH	place						
curing in the oven	open door place close door												
part inspection	inspect	check part	inspect			quality scan	inspect			quality check	inspect	quality check	inspect
Place in finished bin	pick place			place pallets on conveyor	pick place	send PCB to next station	pick place						

Generic Manufacturing Process		
Generic Task Groups	Generic Tasks	Generic Skills
Input Material Handling	load material to work space check material	scan match pick place push pull turn
Product Transfor-mation	apply procedure to finalize product	move follow pick place open close
Quality Control	check product quality	inspect
Output Product Handling	sort final product	pick place

Figure 46. Skills Analysis on Five Industrial Pilot Lines

7.3.2.1.2 Define classes and their properties.

In addition to studying the definition of terms in literature and analyzing the usage of the terms in the ACROBA use cases, the model of Pedersen et al. [137] also contributed to the decision of the classes. Pedersen et al. presented their model with three abstraction levels: task, skill, and primitive [137]. This model is commonly used in the robotics industry. The main classes are based on [137] with modifications and extensions. The following classes are decided for the ontology, and they are described here in natural language.

Robotic Process: this is a process that involves robots in production. Robot Process is a process. It has at least one Robot as Agent participates in. A Robot Process consists of at least one Robot Task. It can have human tasks, robot tasks and human robot collaboration tasks.

Task: It is part of a Robot Process. It has three types: Robt Task, Human Task, Human Robot Collaboration Task.

Robot Task is defined as a compound activity that is done by robots to perform complex robot actions. It consists of robot skills.

Human Task is an activity that is performed by a human as an agent. It has at least one output to the system that indicates the Human Task is finished.

Human Robot Collaboration Task (HRC Task) is a compound activity that is performed by both human and robot to achieve a common goal. There are five types of HRC tasks: Coexistence Fence (CF), Sequential Cooperation SMS (SS), Teaching HG (TH), Parallel Cooperation SSM (PS), and Collaboration PFL (CP) [152].

Robot Skill (in short Skill) is classified as generic skills and application skills. Generic skills are skills that can be reused by every application. Application skills are created just for the usage in one specific application. Robot Skills are defined as intuitive robot abilities for performing an autonomous mission. Skills should be easily understood by non-experts for

parameterization. A skill is made up of a combination of primitives and/or other skills. Skills are used to complete higher level tasks [137].

Robot Primitive is defined as an atomic robot action for performing a single operation. Primitives can be combined to form more complex robot actions - Robot Skills. They are the lowest level model elements and provide a direct link to hardware elements. Primitives are not hardware dependent. Normally, the same type of device provides the same primitives, therefore the devices within the same type should be interchangeable without affecting the control layer. For example, all grippers can "Grasp" and "Release". Primitives are mapped to executable codes on the robot system to manage the low-level robot controls.

KPI: Key performance indicators (KPIs) are set of metrics evaluating performance of processes or tasks. KPIs specify Requirement. KPI represents quantitative measures of Critical Success Factors (CSFs) in enterprises. KPIs play an essential role in seeking improvements and reaching better performance.

Requirement is defined as the target goal of a process/subprocess/task. Each requirement can have one or more KPIs evaluating its achievements. A Requirement can have multiple KPIs.

The properties, conditions, and cardinality of the terms are defined in Table 10.

Table 10. Class and Properties

Class	Property	Condition	Cardinality
Process	name goal sequence input output feedback precondition postcondition		has at least one task
Task	name sequence input output feedback precondition postcondition		has at least one human task or robot skill
Human task	name human agent output	no robot involvement	has at least 1 human agent
Robot Task	name sequence input output feedback precondition postcondition	no human involvement	no human agent
HRC Task	name sequence inputs outputs feedback precondition postcondition HRC task type HRC mode safety level	human and robot are both involved in completing the activity	has at least 1 human task and 1 robot task

Robot Skill	name sequence input output feedback precondition postcondition	no human involvement	has at least 1 primitive
Robot Primitive	name input output feedback precondition postcondition	has robot as agent	atomic
KPI	name formular		
Requirement	name goal		

7.3.2.2 Formalization

In the process of formalization, the informal terms and axioms that are written in natural language will be described in a formal language. It is recommended to reuse or extend existing ontologies whenever possible, especially for upper ontologies, because it can reach a higher level of maturity and compatibility as well as increase acceptance [7]. Therefore, the intension is to connect the ORPP ontology to the standard ontologies for robotics and automation: CORA (IEEE standard ontology for robotics and automation) [135] and its theoretical foundation SUMO (Suggested Upper Merged Ontology) [136].

CORA is an IEEE standard ontology for robotics and automation. CORA aims to create a common vocabulary/language for robotics and automation domain. It formally defines robots, robot parts, robot groups, robot positions and configurations, robot autonomy levels, and robotic systems [7,142].

SUMO is the Suggested Upper Merged Ontology which is an upper-level ontology that provides definitions of basic ontological concepts for all domains to use as a foundation [136]. To ensure reusability and align with existing standards the ORPP and its supporting ontologies are developed based on CORA [135] and SUMO [136].

Developing ORPP based on CORA and SUMO will ensure reusability and align with existing standards. The ontology architecture is shown in Figure 47. It shows that ORPP consists of two levels of ontologies (in blue): domain ontology and application ontology. The domain ontology “ORPP Core” covers the main robotic process concepts. The application ontologies cover the task planning (PDDL), use case ontology (ACROBA), and the Robot

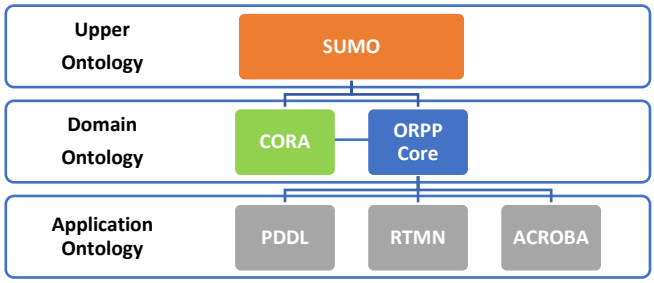


Figure 47. Ontology Architecture.

Task Modelling Notation (RTMN) concepts. PDDL, RTMN and ACROBA ontology are not in scope. They are marked in Grey.

Figure 48 presents the overall concepts based on the ORPP architecture. It is modeled in UML diagram.

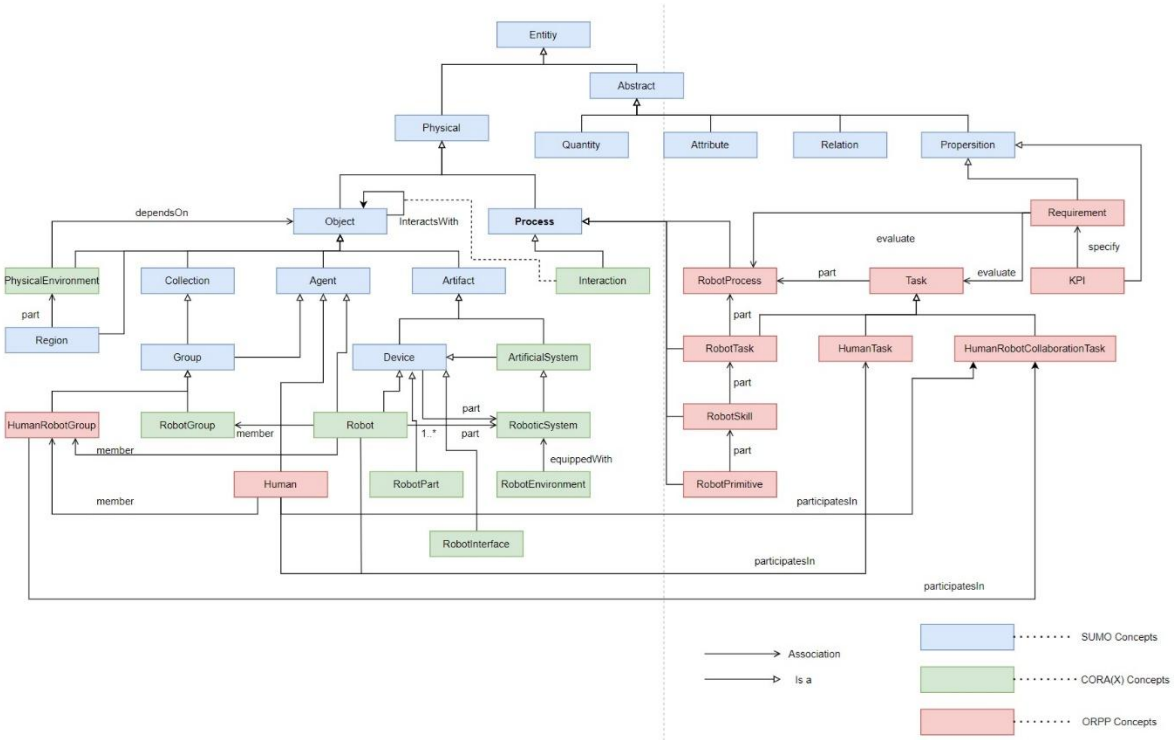


Figure 48. Taxonomy of the main Concepts of ORPP and Their Relation to the Other Ontologies.

7.3.2.2.1 SUMO Concepts

SUMO classes are shown in Figure 48 in blue. Entity is the top SUMO category. It is the universal class of individuals. It has two disjoint subclasses, Physical and Abstract. The definitions are quoted from [135].

Physical: "An entity that has a location in space-time."

Abstract: "Properties or qualities as distinguished from any embodiment of the properties/qualities in a physical medium. Instances of Abstract can be said to exist in the same sense as mathematical objects such as sets and relations, but they cannot exist at a particular place and time without some physical encoding or embodiment."

Process: "The class of things that happen and have temporal parts or stages. Examples include extended events like a football match or a race, actions like pursuing and reading, and biological processes. The formal definition is anything that occurs in time but is not an object."

Object: "Corresponds roughly to the class of ordinary objects. Examples include normal physical objects, geographical regions, locations of processes, and the complement of objects in the physical class."

Region: "A topographic location. Regions encompass surfaces of objects, imaginary places, and geographic areas. Note that a region is the only kind of object that can be located at itself."

Collection: "Collections have members like classes, but, unlike classes, they have a position in space-time and members can be added and subtracted without thereby changing the identity of the collection."

Agent: "Something or someone that can act on its own and produce changes in the world.

Artifact: "An object that is the product of a making."

Group: "A collection of agents."

Device: "A device is an artifact whose purpose is to serve as an instrument in a specific subclass of a process."

Quantity: "Any specification of how many or how much of something there is. Accordingly, there are two subclasses of quantity: number (how many) and physical quantity (how much)."

Attribute: "Qualities that we cannot or choose not to reify into subclasses of object."

Set or Class: "The set or class of sets and classes, i.e., any instance of abstract that has elements or instances."

Relation: "The class of relations. There are three kinds of relation: predicate, function, and list. Predicates and functions both denote sets of ordered n-tuples. The difference between these two classes is that predicates cover formula-forming operators, while functions cover term-forming operators. A list, on the other hand, is a particular ordered n-tuple."

Proposition: "Propositions are abstract entities that express a complete thought or a set of such thoughts."

7.3.2.2.2 CORA(X) Concepts

CORA and its supporting ontologies classes are shown in Figure 48 in Green. The definitions are quoted from [135].

Device: "Device is partitioned to Robot, Robot Part, Robot Interface, and Electric Device. Robots have other devices as parts."

Robot: "a robot is a Device (SUMO term) which participates as a tool in a process. A robot is also an Agent which is something that can act on its own and produce changes. Robots perform tasks by acting on the environment/themselves."

RobotPart: "RobotPart is further divided into robotActuatingPart, robotCommunicatingPart, robotProcessingPart and robotSensingPart."

RobotInterface: "Robot interacts with the world surrounding it through an interface. RobotInterface is a device composed of other devices that play the roles of sensing device, actuating device and communicating device. Through the interface, the robot can sense and act on the environment as well as communicate with other agents. Therefore, the robot interface can be viewed to refer to all the devices that allow the robot to interact with the world. Every robot interface must have a part that is either a robot sensing part, or a robot actuating part or a robot communicating part."

RobotGroup: “A robot is an agent, and agents can form social groups. According to SUMO, a group is “a collection of agents,” like a pack of animals, a society, or an organization. A RobotGroup is a group whose only members are robots.”

ArtificialSystem: “ArtificialSystem is an Artifact formed by various devices (and other objects) that interact in order to execute a function. For any part of an artificial system, there is at least one other part it interacts with.”

RoboticSystem: “Robots and other devices can form a RoboticSystem. A RoboticSystem is an artificial system formed by robots and devices intended to support robots to carry on their tasks. Robotic systems might have only one or more than one robot.”

Interaction (CORAX): “An interaction is a Process in which two agents participate. It is composed of two subprocesses defining action and reaction. The action subprocess initiated by agent x on a patient agent y causes a reaction subprocess having y as agent and x as patient.”

7.3.2.2.3 ORPP Core Concepts and Axioms

The ORPP core is based on the model of Pedersen et al. [137] with modifications and extensions. Pedersen et al. presented their model with three abstraction levels: task, skill, and primitive [137]. The ORPP model has one more layer - the process layer. It is a very important part as it integrates the tasks and enables the requirement evaluation with KPIs. A process can be broken down into different levels - the primitive layer being the lowest abstraction level which consists of premade programs to control robot actions. The skill layer is made up of generic and specific skills. The task layer consists of human tasks, robot tasks, and HRC tasks. These layers can provide traceability of the KPIs to robot primitives. They allow the KPIs to be calculated automatically in the robotic system. The contribution of the model has the following features:

- It adds a new process layer. It groups the robotic tasks together and provides an overview for the management with definable requirements/goals and KPIs. The process is the starting point of any requirement engineering analysis; it is highly important from the modelling perspective. Since one can track the process down to the primitive level, each robotic action is registered in the system and the KPIs can be calculated based on the information from the primitive layer. These KPIs then provide the evaluation of the requirements/goals.
- It extends the task layer to three task types (Robot Task, Human Task and HRC Task). Separating human and HRC tasks enables the application of different levels of safety controls.
- It enriches skill layers with skills themselves, meaning that skills can be created by not only primitives but also by skills. This enables the reuse of skills that are already created.

In this section the first order logic is used to describe the informal terms and axioms in the ORPP ontology that are written in natural language in the former chapter.

RobotProcess: RobotProcess is a Process. It has at least one Robot as Agent participates in. A RobotProcess consists of at least one RobotTask. It can be composed of HumanTasks, RobotTasks and HumanRobotCollaborationTasks.

```
(subclass   RobotProcess   Process)
```

```
(=>
  (and (RobotProcess ?x)
       (exists (?agent)
               (and (Agent ?agent)
                    (participatesIn ?agent ?x)
                    (Robot ?agent)))
       (exists (?task)
               (and (part ?task ?x)
                    (RobotTask ?task))
       (exists (?task)
               (and (part ?task ?x)
                    (or (HumanTask ?task)
                        (RobotTask ?task)
                        (HumanRobotCollaborationTask ?task)))))
  (instance ?x Process))
```

Task: Task is a Process. It is part of a RobotProcess. It has subclass RobtTask, HumanTask, HumanRobotColloaborationTask.

```
(subclass   Task   Process)

(=>
  (and (Task ?x)
       (part ?x ?robotProcess)
       (or (RobotTask ?x)
           (HumanTask ?x)
           (HumanRobotCollaborationTask ?x)))
  (instance ?x Process))
```

RobotTask: RobotTask is a Task. It is part of a RobotProcess. It has at least one Robot as Agent. It cannot have Human as Agent. It consists of RobotSkills.

```
(subclass   RobotTask   Task)

(=>
  (and (RobotTask ?x)
       (partOf ?x ?robotProcess)
       (exists (?agent)
               (and (Agent ?agent)
                    (participatesIn ?agent ?robotProcess)
                    (Robot ?agent)))
       (not (exists (?agent)
                    (and (Agent ?agent)
                         (participatesIn ?agent ?robotProcess)
                         (Human ?agent))))
       (exists (?skill)
               (and (part ?skill ?x)
                    (RobotSkill ?skill))))
  (and (Task ?x)
       (instance ?x Process)))
```

HumanTask: HumanTask is a Task. It is part of a RobotProcess. It has at least one Human as Agent. It cannot have Robot as Agent.

```
(subclass   HumanTask   Task)

(=>
  (and (HumanTask ?x)
       (partOf ?x ?robotProcess)
       (exists (?agent)
               (and (Agent ?agent)
                    (participatesIn ?agent ?robotProcess)
                    (Human ?agent)))
       (not (exists (?agent)
                    (and (Agent ?agent)
                         (participatesIn ?agent ?robotProcess)
                         (Robot ?agent)))))
```

```
  (and (Task ?x)
       (instance ?x Process)))
```

HumanRobotCollaborationTask (HRC Task). HRCTask is a Task. It is part of a RobotProcess. It has at least one Human and one Robot as Agent. There are five types of HRC tasks: Coexistence Fence (CF), Sequential Cooperation SMS (SS), Teaching HG (TH), Parallel Cooperation SSM (PS), and Collaboration PFL (CP).

```
(subclass   HumanRobotCollaborationTask   Task)

(=>
  (and (HumanRobotCollaborationTask ?x)
       (partOf ?x ?robotProcess)
       (exists (?human ?robot)
               (and (Agent ?human)
                    (Agent ?robot)
                        (participatesIn ?human ?robotProcess)
                    (participatesIn ?robot ?robotProcess)
                    (Human ?human)
                    (Robot ?robot))))
  (and (Task ?x)
       (instance ?x Process)))
```

RobotSkill: RobotSkill is a Process. It is part of a RobotTask. It can also be part of a RobotSkill. It consists of Primitives and/or other RobotSkills. RobotSkill has Robot as its Agent. Human cannot be the Agent of RobotSkill.

```
(subclass   RobotSkill   Process)

(=>
  (and (RobotSkill ?x)
       (partOf ?x ?taskOrSkill)
       (exists (?agent)
               (and (Agent ?agent)
                    (participatesIn ?agent ?taskOrSkill)
                    (Robot ?agent)))
       (not (exists (?agent)
                    (and (Agent ?agent)
                         (participatesIn ?agent ?taskOrSkill)
                         (Human ?agent))))
       (exists (?primitiveOrSkill)
               (and (part ?primitiveOrSkill ?x)
                    (or (Primitive ?primitiveOrSkill)
                        (RobotSkill ?primitiveOrSkill)))))
  (and (Process ?x)
       (instance ?x Process)))
```

RobotPrimitive: RobotPrimitive is a Process. It is part of a RobotSkill. RobotPrimitive is atomic. RobotPrimitives has Robot or RobotPart as Agent. Human cannot be the Agent of RobotPrimitive.

```
(subclass   RobotPrimitive   Process)

(=>
  (and (RobotPrimitive ?x)
       (partOf ?x ?skill)
       (exists (?agent)
               (and (Agent ?agent)
                    (participatesIn ?agent ?skill)
                    (or (Robot ?agent)
                        (RobotPart ?agent)))
               (not (Human ?agent))))
  (and (Process ?x)
       (instance ?x Process)))
```

HumanRobotGroup: A Robot or a Human is an Agent, and agents can form social groups. According to SUMO, a group is "a collection of agents,". A HumanRobotGroup is a group that has human and robots as members. It participates in HumanRobotCollaborationTasks.

```
(subclass   HumanRobotGroup   Group)

(=>
  (and (HumanRobotGroup ?group)
       (exists (?member)
               (and (Agent ?member)
                    (memberOf ?member ?group)
                    (or (Human ?member)
                        (Robot ?member)))))
  (instance ?group Group))
```

Requirement: Requirement is a subclass of Proposition. Requirements evaluate Task or Process. Requirement has KPI to specify it. A Task or Process can have multiple Requirements.

```
(subclass   Requirement   Proposition)

(=>
  (and (Requirement ?req)
       (or (evaluates ?req ?task)
           (evaluates ?req ?process)))
  (instance ?req Proposition))
```

KPI: KPI is a subclass of Proposition. KPIs specify Requirement. A Requirement can have multiple KPIs. KPI and Requirement are disjoint subclass of Proposition.

```
(subclass   KPI   Proposition)

(=>
  (and (KPI ?kpi)
       (specifies ?kpi ?requirement))
  (and (subclass ?kpi Proposition)
       (subclass ?requirement Requirement)
       (disjointSubclass KPI Requirement)))
```

Human: Human is an Agent who can act on his own and produce changes. It participates in HumanTask or HumanRobotCollaborationTask. Humans perform these tasks by acting on the environment/themselves.

```
(subclass   Human   Agent)

(=>
  (and (Human ?human)
       (participatesIn ?human ?task)
       (or (instance ?task HumanTask)
           (instance ?task HumanRobotCollaborationTask))
       (actsOn ?human ?environment))
  (and (Agent ?human)
       (canActOnOwn ?human)
       (producesChanges ?human)
       (instance ?human Human)))
```

7.3.2.2.4 PDDL Concepts

The PDDL definition covers domain description, problem description, predicates, actions, initial state, and goal state. The PDDL definitions can be described in an ontology. To use ontology to support the PDDL, a mapping between the ontology elements and the PDDL constructs must be established. Such an ontology can support the PDDL by providing a structured and semantically rich representation of the domain concepts and relationships.

The mapping includes the following:

- Types in the PDDL are represented as classes in the PDDL ontology.
- Objects are instances of types in the PDDL, and they are mapped to ontology to individuals.
- The PDDL uses predicates to define relationships between objects, and these are described as object properties in ontology.
- Initial state specifies the state of the world before the start in the PDDL. Ontology uses object property assertions to represent these individual instances. Individual instances in the ontology are associated with their properties using object property assertions. It is the same for the goal state.
- Actions in the PDDL represent possible transitions between states. They can be created as classes and properties in ontology.
- Constraints are rules and limitations for actions in PDDL. In oncology, they are described using additional axioms and properties.

The main concepts of the PDDL application ontology are Plan, Action, InitialState, GoalState, CurrentState, Domain, and Problem. The PDDL ontology is under development, and the basic concept is shown in Chapter 7 in the PDDL application scenario. The complete PDDL ontology will be presented after validation.

7.3.2.2.5 ACROBA Concepts

The ACROBA uses case ontology covers five industrial manufacturing use cases (medical device, large plastic, electric motor, and electronic components) in the ACROBA project [1]. As all use cases are different in terms of products and processes, the use case-specific knowledge in the application ontology is separated from the ORPP core because it allows the ORPP to cover only the generic concepts that lead to flexibility and reusability. The main concepts of the ACROBA ontology are as follows: Organization, User, Role, Product, ProductPart, ProductionPlan, and Device. This application ontology is under development and will be presented when it is finished.

7.3.2.2.6 RTMN Concepts

RTMN [151] is a graphical modelling language that enables the users to intuitively model their production processes and execute them. As it is used for developing a Graphical User Interface (GUI), it is considered an application ontology. The author developed RTMN in her previous research, and it is customized and aligned with their general approach for robot task planning. The concept used in RTMN will be modeled as classes and object properties in the ORPP ontology. This ontology enables the GUI generation. The RTMN concepts include

Process, Task, Skill, Primitive, Sequence Flow, Condition, Decision, Start/End, and Notation. The RTMN is under development and will be presented when it is validated.

7.4 ONTOLOGY IMPLEMENTATION (PHASE 4)

The ORPP core ontology was implemented in Protégé 5.5.0 [157] because Protégé is the well-known open-source ontology software in this field. Protégé allows the user to save the ontology in a variety of OWL formats. The terms in ORPP are defined as Classes in Protégé. Relationships are defined using Object Property. Data Property defines data type properties. Individuals are the instantiations of the classes and properties.

Figure 49 is the relation graph of the implemented ontology. It shows the relationships between classes, object properties and instances.

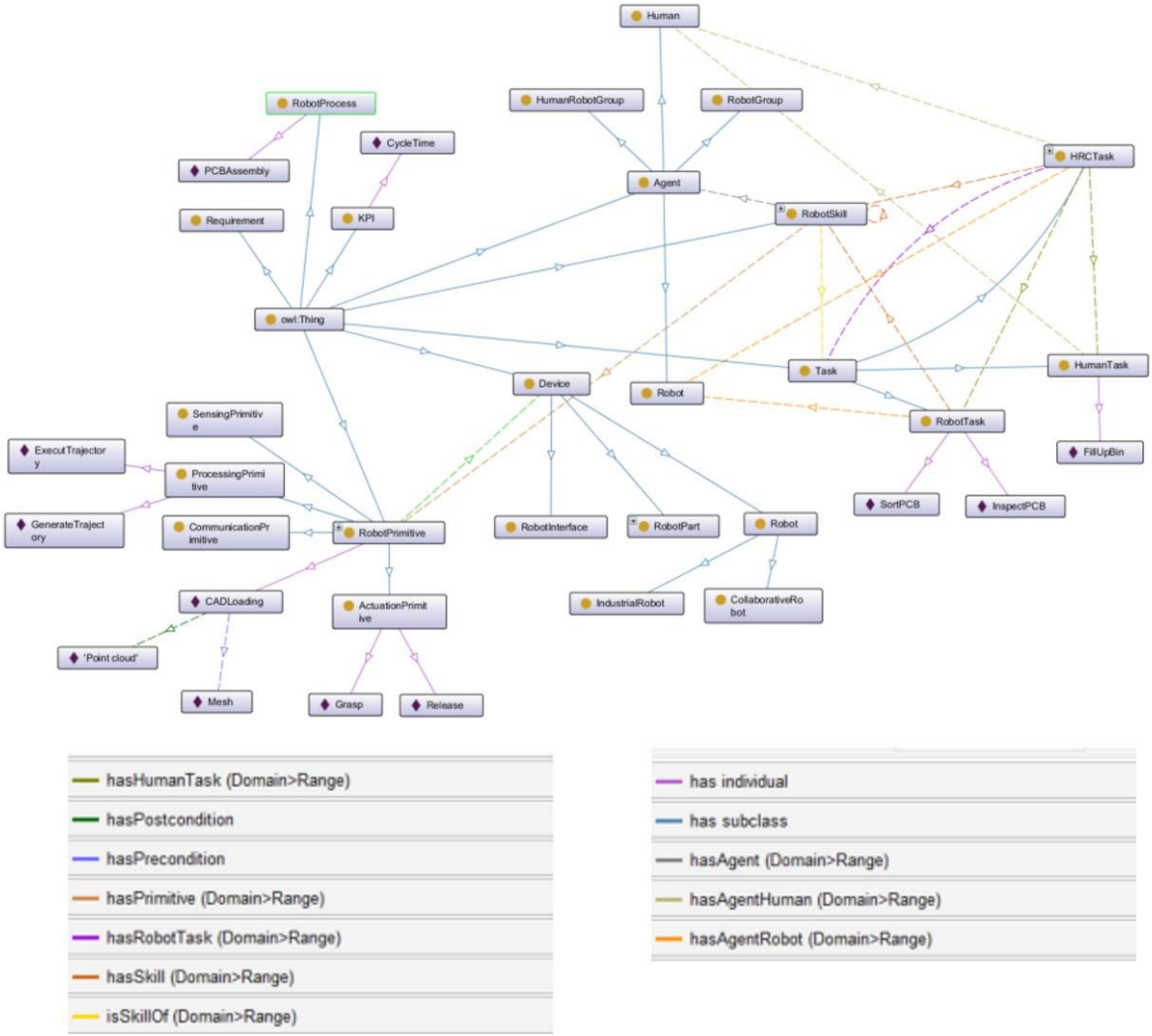


Figure 49. ORPP Core Relation Graph

In Figure 49 the orange nodes are classes, and the blue arrows indicate they have subclasses. Relationships between classes are modeled as object properties in Protégé. Here, some of the most important classes and their related properties are described.

The Process Class has object properties: hasTask (process consists of tasks), isEvaluatedby (process is evaluated by requirements). Data properties include Input, Output, Feedback, Precondition, and Postcondition.

The Task Class has the following object properties: hasAgent (one can assign an agent to the task) and hasSkill. It has data properties: Input (e.g., 3D coordinates), Output (e.g., grasping pose), Preconditions (e.g., robot at A), Postconditions (e.g., robot at B), Feedback, and Sequence (a list of tasks to be performed in this task). Task has three subclasses: Robot Task, Human Task and Human–Robot Collaboration Task (HRC Task). HumanTask has a human agent. RobotTask has a robot agent. The HRC Task has human and robot agents. It also has DistanceHumanRobot (the distance between the human and the robot), HumanPosition (the human x, y, z location), RobotPosition (the robot x, y, z location), and RobotSpeed (the speed of the robot) as data properties.

The Skill Class has the following main object properties: hasAgent (Robot), hasPrimitive (Skill consists of primitives), hasSkill (skills consists of other skills), and isSkillof (Task and skill contain skills, skill is a skill of a skill or task). isSkillof is the inverse property of hasSkill. Data properties include Sequence, Input, Output, Feedback (e.g., after running the skill, it returns an image from a camera), Precondition, and Postcondition.

The Primitive Class has the following main object properties: isPrimitiveof (which is the inverse property of hasPrimitive). Data properties include Input, Output, Feedback, Precondition, and Postcondition.

The Requirement Class has the following object properties: isSpecifiedby (requirements are specified by KPIs) and evaluates (requirement evaluates processes), and it is the inverse property of isEvaluatedby. The KPI Class has the following object properties: specify (KPI specifies requirements).

7.5 Ontology Testing (Phase 5) - The Use of ORPP in the ACROBA Project

In the ACROBA project, the ORPP ontologies are used in three ways: 1. populating the GUI; 2. supporting task execution; 3. supporting task planning. The overview of the usage is shown in Figure 50.

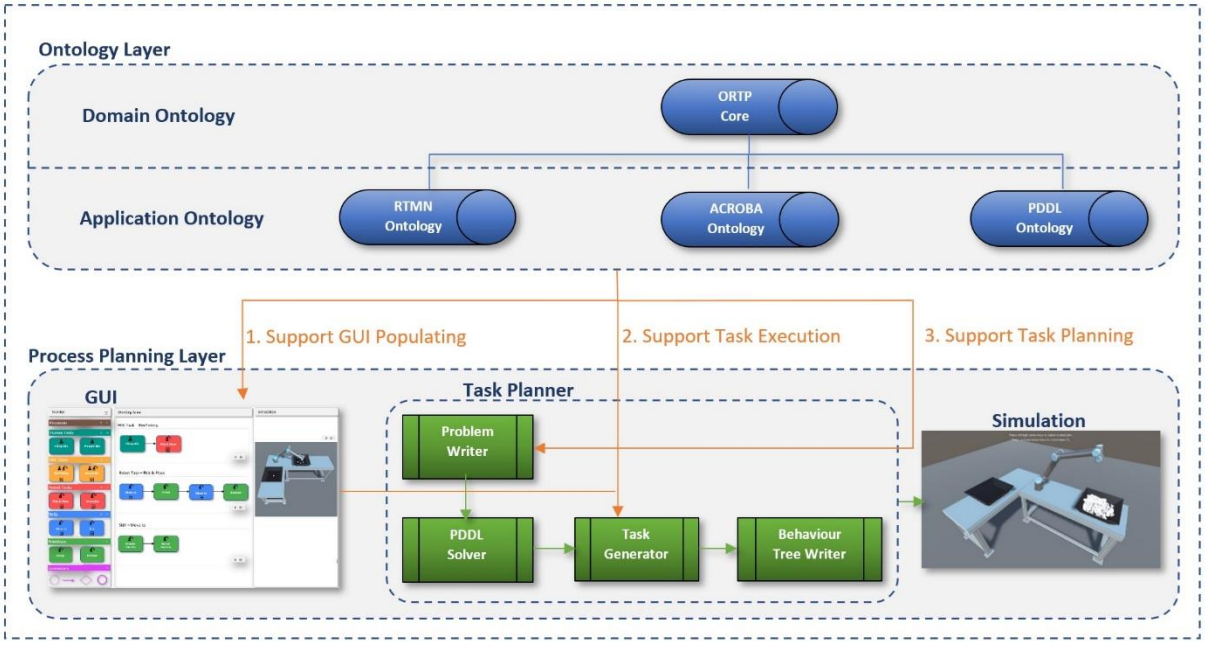


Figure 50. Uses of ORPP ontologies

7.5.1 Populating the GUI (Orange Arrow 1.)

The RTMN graphical user interface allows users to drag and drop tasks, skills, and primitives to compose a manufacturing process. To be able to load this information correctly in the GUI, ontology is used to generate the GUI sections with the correct processes, tasks, skills, and primitives. Users can log into the populated GUI and use the drag and drop function to model a process. To create a process model, one needs to first create the different tasks in the process, presuming skill and primitives are already available in the GUI. Then, one can use these created tasks to compose the process. The instances of the process model are saved back to the ontology as individuals. New robot skills can be created based on existing primitives and other skills. For example, the robot skill Pick is created by the robot skill Move and robot primitive Grasp.

7.5.2 Supporting task execution (Orange Arrow 2.)

Process is executed by task sequences. Task sequence is given in the RTMN process model. ACROBA executes task sequences based on the core ORPP classes and relationships. Each task has a sequence of skills. For example, a “pick and place” task has the skill sequence Move, Match, Pick, Move, and Place assigned to it. Each skill has a primitive sequence. By launching a process, the sequences of tasks, skills and primitives are launched. These sequences are sent to the task planner (a module of ACROBA platform) and translated further into behavior threes.

7.5.3 Supporting PDDL task planning (Orange Arrow 3.)

ACROBA ontology defines all the five use cases—their organization, users, their products, and processes. It supports the PDDL task planning by providing the domain and problem definitions in the ACROBA ontology. The PDDL planning aims to generate a skill sequence that solves a given task. The task planner translates all relevant information from the ontology into the PDDL [153] domain and problem representation. For example, a skill in the ontology is converted into an action in the PDDL [147].

7.5.4 PCB Assembly Application - Example of Orange Arrow 1&2

The testing of the ontology is based on a real-world scenario in the ACROBA project. The instantiation covers one simple assembly process. It is illustrated in Figure 51 - a Printed Circuit Board (PCB) assembly process (this process is created based on the IKOR process, See Appendix 15.1 for information related to IKOR).

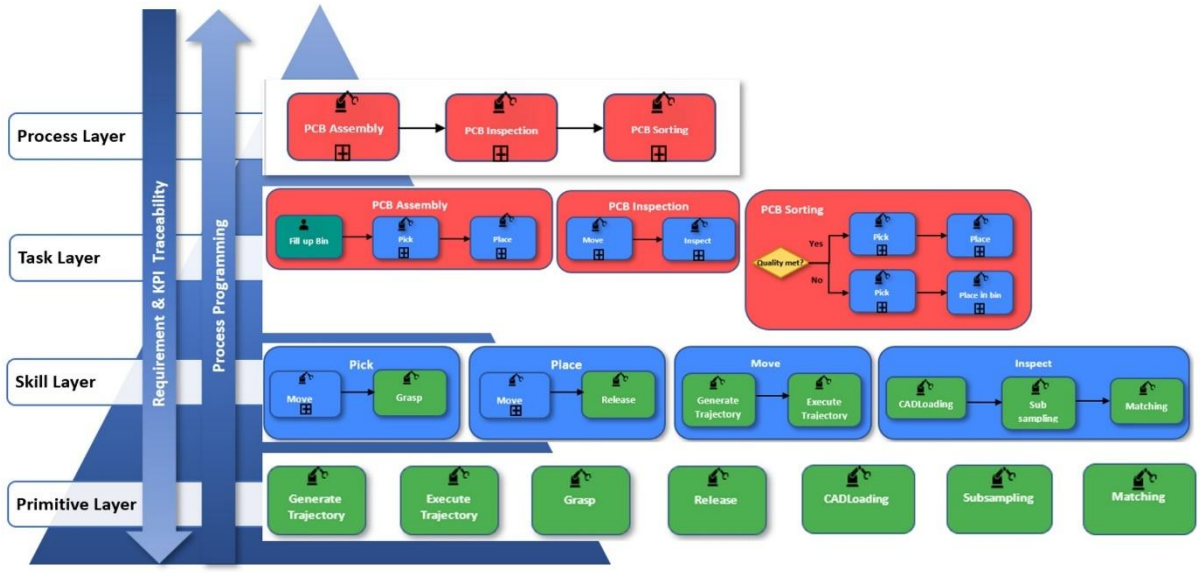


Figure 51. Projection of an ontology instantiation—a PCB assembly process

The PCB is assembled by putting different kinds of pin through hole (PTHs) components on the board. The process is called PCBAssembly as an instantiation of the class process. This PCBAssembly process has three tasks: PCB assembly (AssemblePCB: an instantiation of a HRCTask), PCB inspection (InspectPCB: an instantiation of a RobotTask) and PCB sorting (SortPCB: an instantiation of a RobotTask). PCB assembly is conducted by a HumanTask - FillUpBin (bin is used as the container of the PTHs) and two RobotSkills - Pick and Place. The PCB inspection task is undertaken by RobotSkills - Move and Inspect. The PCB sorting uses two RobotSkills - Pick and Place - to sort the PCB at different places.

Figure 52 shows that HRCTask has an instance, AssemblePCB. It is part of the PCBAssembly process instantiation. This specific task has a Human called Ron and a UR5 Robot as its Agents. It has HumanTask FillUpBin instantiation and two RobotSkills, Pick and Place. It has a sequence and some other uses in Figure 52.

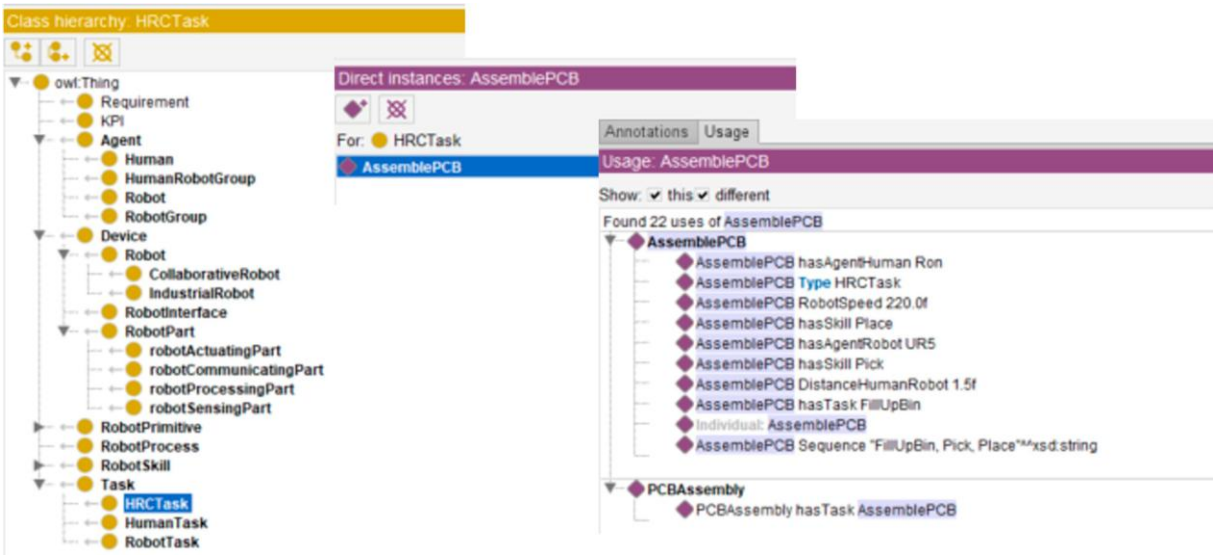


Figure 52. Individual HRCTask and its use in ORPP.

Protégé has integrated functionalities to run SPARQL (short for SPARQL Protocol and RDF Query Language) queries. SPARQL is an RDF query language and protocol produced by the W3C RDF Data Access Working Group (DAWG). It is used to create queries to get information/knowledge from ontology. These queries provide answers to the ontology users to support them with decision making. Figure 53 is an example of a query to return the instantiation of the PCBAssembly process.

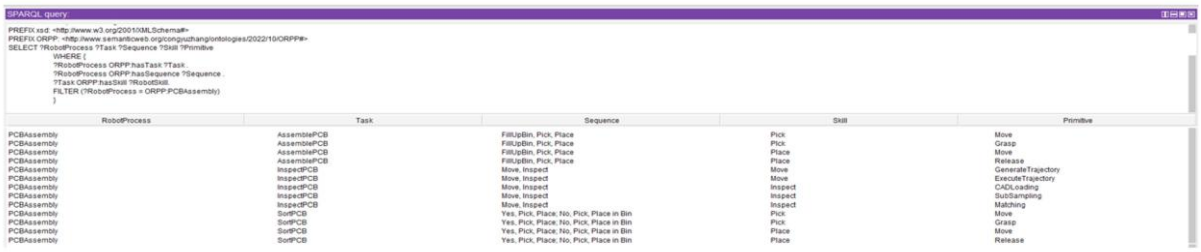


Figure 53. SPARQL query for PCBAssembly process instantiation.

The instances of the Process, Task, Skill, and Primitives can be used to populate the GUI. The sequences are used to support task execution. Here, only the ORPP ontology is used. In this example, if the ACROBA, RTMN, and PDDL ontologies are ready, one can link the information and have a much more comprehensive ontology to be queried for complex planning scenarios. See Figure 54 for the information use.

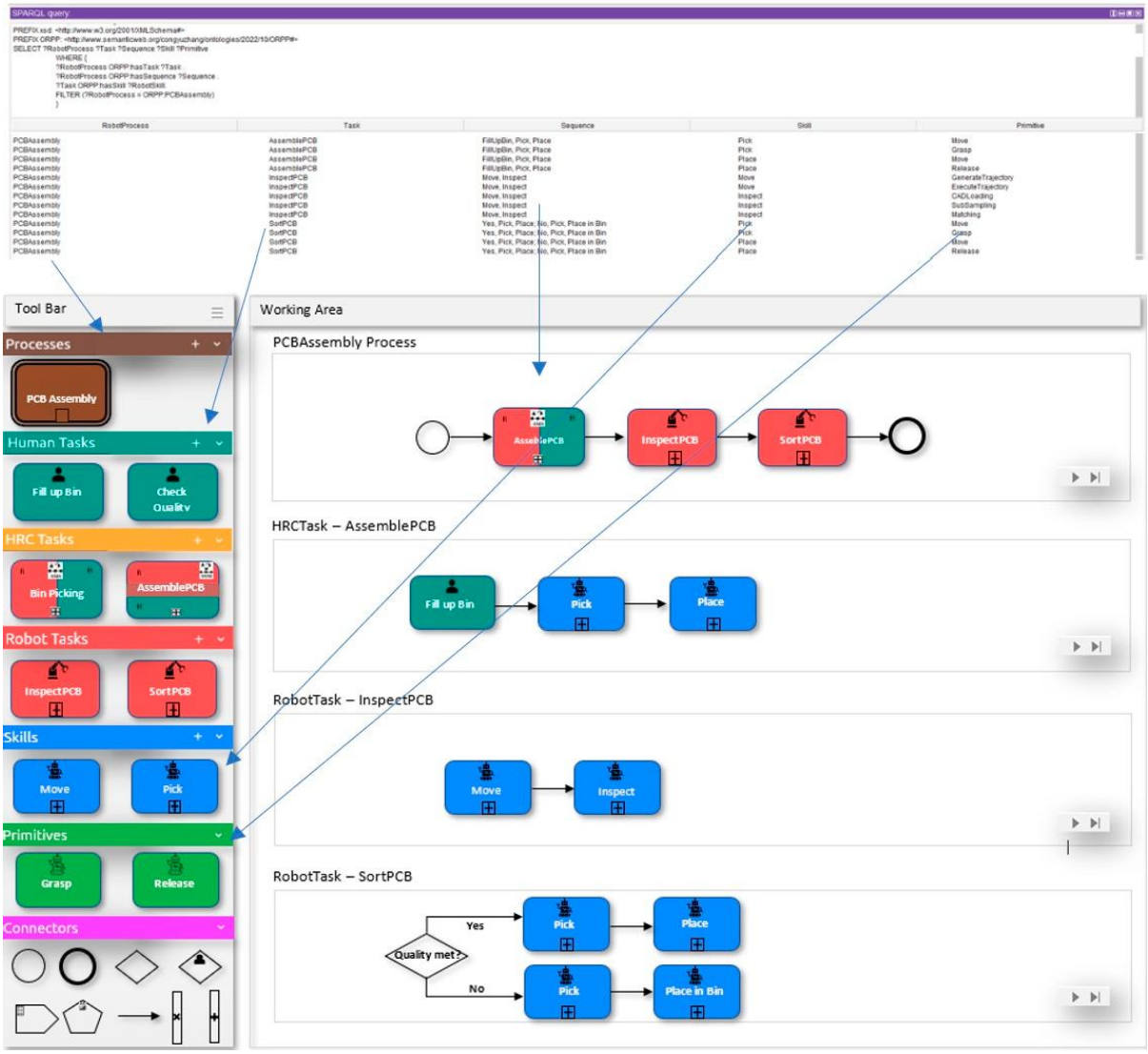


Figure 54. Ontology Query Support Use of Ontology

The ontology queries can answer the following competency questions: CQ1, CQ4, CQ5, CQ7, CQ10, CQ11, CQ12, CQ14, and CQ16.

7.5.5 PDDL Task Planning Application- Example of Orange Arrow 3

7.5.5.1 Cell Setup

Figure 55 is a picture of the demonstrator. It shows the setup of the cell. This cell consists of the following components.

- Robot: UR5e
- Gripper: ROBOTIQ 2F-85
- Camera: Zivid M+

- There is a “grid” that has three by three holes, and their positions are (00,01,02, 10,11,12,20,21,22)
- There are “cylinder” and “circular-container”. A cylinder can be put into a circular-container. Both cylinders and circular-containers have three colors: red, yellow, and green.
- There are three bins for the cylinder (they are bin red, bin yellow, and bin green), and each bin contains six cylinders. Each bin contains one color of the cylinders
- There are three hangers for the circular-container (they are hanger red, hanger yellow, and hanger green), and each hanger contains one color of circular-containers

Figure 55. The Demonstrator

7.5.5.2 The Scenario

The task is to assemble the cylinders and circular-container in the grid. The assembly goal is to put cylinders into circular-containers on all holes of the grid with matching colors. In other words, the goal of this scenario is for robots to move cylinders and circular-containers from the initial state to the goal state.

- The Initial State

The user decides and sets the initial state for all positions: (00, yellow circular-container, no cylinder), (01, yellow circular-container, no cylinder), (02, red circular-container, no cylinder), (10, yellow circular-container, no cylinder), (11, yellow circular-container, yellow cylinder), (12, red circular-container, no cylinder), (20, green circular-container, no cylinder), (21, green circular-container, green cylinder), and (22, green circular-container, no cylinder). It is presented in Figure 56.

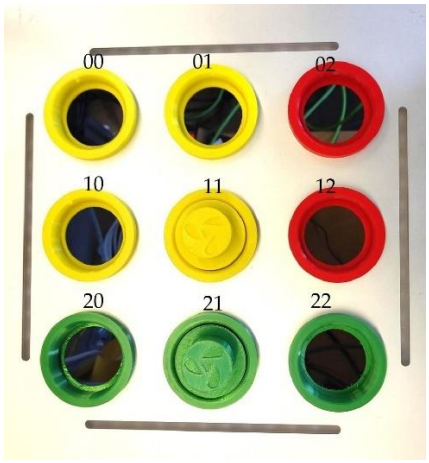


Figure 56. Initial State

- The Goal State

The user sets the goal state (see Figure 57), which is to put cylinders into circular-containers on all holes of the grid with matching colors.

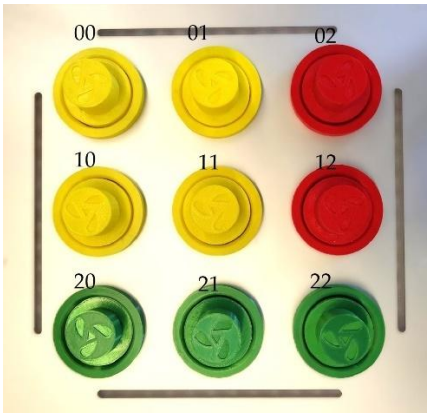


Figure 57. Goal State

The goal state is given for all positions: (00, yellow circular-container, yellow cylinder), (01, yellow circular-container, yellow cylinder), (02, red circular-container, red cylinder), (10, yellow circular-container, yellow cylinder), (11, yellow circular-container, yellow cylinder), (12, red circular-container, red cylinder), (20, green circular-container, green cylinder), (21, green circular-container, green cylinder), and (22, green circular-container, green cylinder).

- The Constrains

The robot can move one thing at a time, one cylinder can be placed in one circular-container, and a circular-container cannot be put on a cylinder. Circular-containers can be put in holes on the grid; cylinders are not allowed to be put in holes.

7.5.5.3 The PDDL (Planning Domain Definition Language) Definitions

The PDDL definition covers: domain description, problem description, predicates, and actions definitions. It includes the constraints described above, the ability to move cylinders between circular-containers, placing circular-containers on the grid, and ensuring color-

matching between cylinders and circular-containers. The initial state and goal state are defined based on the scenario described earlier.

- PDDL Domain Description

The PDDL code of the domain description is shown below. For the PDDL planning to work, more complex descriptions are needed. Here, a simplified version is presented.

```
(define (domain assembly-dom)
(:types
     hole cylinder circular-container bin color
)
(:predicates
     (hasColor ?obj - cylinder ?color - color)
     (contains ?bin - bin ?cylinder - cylinder)
     (empty ?bin - bin)
     (onGrid ?container - circular-container ?hole - hole)
     (placed ?cylinder - cylinder ?container - circular-container)
     (goal-state-reached)
)
(:action pick
     :parameters (?cylinder - cylinder ?bin - bin)
     :precondition (and (contains ?bin ?cylinder) (empty ?cylinder))
     :effect (and (not (contains ?bin ?cylinder)) (not (empty ?cylinder)))
)
(:action place
     :parameters (?cylinder - cylinder ?container - circular-container ?hole - hole)
     :precondition (and (empty ?container) (onGrid ?container ?hole) (not (placed ?cylinder
     ?container)))
     :effect (and (not (empty ?container)) (placed ?cylinder ?container)))
)
```

- PDDL Problem Description

The basic code of the problem description is shown below. This, too, is a simplified version of the problem description used to illustrate the ontology support for the PDDL.

```
(define (problem assembly-prob) (:domain assembly-dom)
(:objects
     h00 h01 h02 h10 h11 h12 h20 h21 h22 - hole
     c1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15 c16 c17 c18 - cylinder
     cc00 cc01 cc02 cc10 cc11 cc12 cc20 cc21 cc22 - circular-container
     bin-red bin-yellow bin-green - bin
     red yellow green - color
)
(:init
     ; initial state of bins
     (contains bin-red c1) (hasColor c1 red)
     (contains bin-red c2) (hasColor c2 red)
     (contains bin-red c3) (hasColor c3 red)
     (contains bin-red c4) (hasColor c4 red)
     (contains bin-red c5) (hasColor c5 red)
     (contains bin-red c6) (hasColor c6 red)
     (contains bin-yellow c7) (hasColor c7 yellow)
     (contains bin-yellow c8) (hasColor c8 yellow)
     (contains bin-yellow c9) (hasColor c9 yellow)
     (contains bin-yellow c10) (hasColor c10 yellow)
     (contains bin-yellow c11) (hasColor c11 yellow)
     (contains bin-green c13) (hasColor c13 green)
     (contains bin-green c14) (hasColor c14 green)
     (contains bin-green c15) (hasColor c15 green)
     (contains bin-green c16) (hasColor c16 green)
     (contains bin-green c17) (hasColor c17 green)
     ; Initial state of circular containers on the grid
     (onGrid cc00 h00) (hasColor cc00 yellow)
     (onGrid cc01 h01) (hasColor cc01 yellow)
     (onGrid cc02 h02) (hasColor cc02 red)
     (onGrid cc10 h10) (hasColor cc10 yellow)
     (onGrid cc11 h11) (hasColor cc11 yellow)
     (onGrid cc12 h12) (hasColor cc12 red)
     (onGrid cc20 h20) (hasColor cc20 green)
     (onGrid cc21 h21) (hasColor cc21 green)
     (onGrid cc22 h22) (hasColor cc22 green)
     ; Initial state of cylinders in circular containers
```

```
        (placed c12 cc11) (placed c18 cc21)
        ; Initial state of bins
        (not (empty bin-red)) (not (empty bin-yellow)) (not (empty bin-green))
    )
    (:goal (and
        (placed c7 cc00) (placed c8 cc01) (placed c1 cc02)
        (placed c9 cc10) (placed c12 cc11) (placed c2 cc12)
        (placed c13 cc20) (placed c18 cc21) (placed c14 cc22)
        (goal-state-reached)
    ))
    )
```

7.5.5.4 Ontology Representation for PDDL Domain and Problem

To use ontology to support the PDDL, a mapping between the ontology elements and the PDDL constructs must be established. This means defining the PDDL concepts with the ontological format. Such an ontology can support the PDDL by providing a structured and semantically rich representation of the domain concepts and relationships.

The ontology representation of the PDDL domain and problem is shown in Figure 58. It is shown as a relation graph in Protégé. This representation is chosen because it provides a graphical overview of the ontology. This ontology is for a specific PDDL domain and problem. To support PDDL planning in general, a more generic ontology needs to be created. The result will be presented when this application ontology is completed and tested.

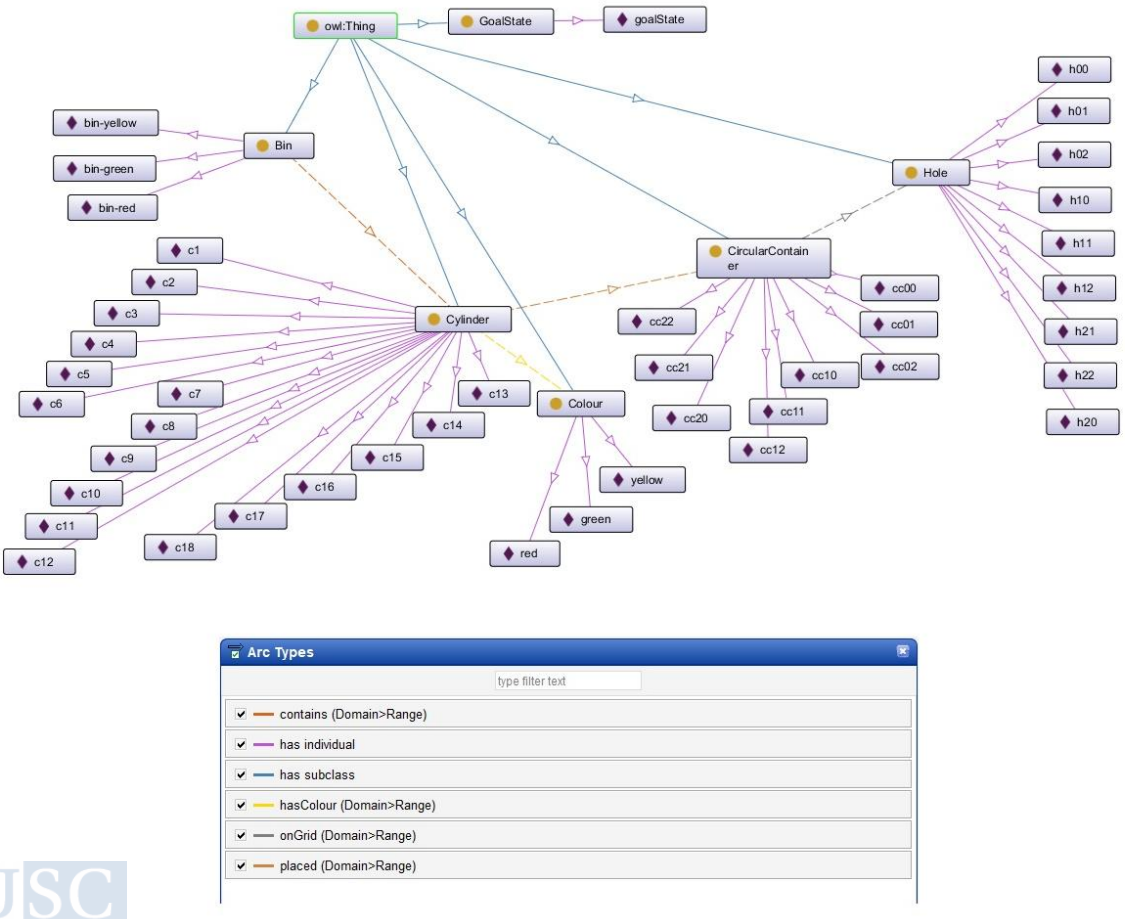


Figure 58. PDDL Ontology Relation Graph

Aligning the PDDL and ontology allows for a more semantically meaningful representation of the plan domain and therefore enables more intelligent planning systems supported by ontological reasoning capabilities. More complex planning scenarios need more expressive and detailed ontology. This scenario does not cover complex scenarios, for instance, allow cylinders placed in the wrong direction (top to bottom) in the initial state, allow random cylinders placed in bins (on top of each other), or allow human robot collaboration.

7.6 CONCLUSION

Robotic process management is not an easy task as it involves numerous tasks. It is advantageous if a robot's knowledge representation combines different types of knowledge with reasoning capabilities, enabling it to devise the optimal plan of action to perform its duties [158]. While a few papers did focus on ontology-based task planning for robotics, further work is required to cover the full agile manufacturing domain. For ontology-based systems to be effective, standardization of the domain is key [146]. In this work, a state-of-the-art analysis of the existing ontology approaches is provided and based on the results, a standardization of the robotic task planning domain is presented. The major gap identified in the literature is the need to bridge the higher-level robotic process planning and the lower-level robotic control. Therefore, an Ontology for Robotic Process Planning (ORPP) was proposed. ORPP covers both industrial and collaborative robots in agile manufacturing. The major contributions of ORPP are as follows: (1) it details the concepts in the robotic manufacturing process and structures the concepts from the process level to the robot control level; (2) it creates traceability of requirements to KPI measurements of the process; (3) it extends the task layer to three task types (Robot Task, Human Task, and HRC Task); (4) it supports the PDDL task planning; (5) it is modular, reusable, and extensible, which facilitates knowledge sharing and reasoning; (6) it aligns with upper-level ontology, extends domain-level ontology, and adds application-level ontology.

This ORPP ontology will be utilized and tested in the ACROBA project. However, the testing of the ontology can only be conducted in a limited way. The limitations are due to the partial implementation of the application ontologies. The PDDL application ontology only covers two scenarios (one in Section 7.5.5 and one in Chapter 9, Section 9.3).

Future research will be focused on Human–Robot Collaboration concepts that are now only covered in the ORPP ontology at a basic level. Richer knowledge about human safety and human factors will be added to extend the current ontologies

8 FLEXIBLE TASK PLANNER USING RTMN2.0 AND ORPP ONTOLOGY

8.1 INTRODUCTION

Chapter 5-7 provides the RTMN for GUI modelling and the ORPP ontology to support and enable robotic process planning and execution. Chapter 8 provides an overview of the complete ACROBA framework, focusing on the task planner. The task planner is the central component of ACROBA. This chapter focuses on how different components of the task planner work together and the synergies between them.

The user requirements for the task planner come from the questionnaires and interviews presented in Chapter 5. After consolidating all the answers, the following requirements are identified:

- The task planner shall offer the functionality for task/process design at the planning phase. This means that an operator who is not a robotic specialist is able to specify the robotic tasks/processes as well as reorder the task/process execution sequence. It is considered a "nice to have" if a task is modifiable at runtime. Indeed, this functionality improves the testing and commissioning times both in simulations with the real robot, by removing the need to rebuild the application when changing the task.
- At runtime, the task planner shall be able to control the execution of the task. If a use case uses an HMI, the task planner shall provide the interface to it. The task planner shall be able to display the current execution of the task and when an error occurs, provide error diagnosis.
- The task planner shall optimize or automate the task/process design through PDDL solving. This shall be implementable in a modular way to achieve flexibility.

Based on these requirements a ROS-based flexible and intelligent task planning system that can be easily adapted to any agile production cell was proposed. This goal has raised several research questions:

- Can existing task planning frameworks fulfil the user requirements?
- What is the best architecture for the task planner?

To answer these questions, the literature in this field was reviewed.

8.2 LITERATURE REVIEW

8.2.1 ROS

ROS1[159] has become a standard framework to develop robotic solutions in the last decade. It is open source and has a wide community of developers and users. It is used both for research and commercial activities [159]. ROS2 intends to become the successor of ROS1, solving important issues like real-time and non-ideal network communication. To do that, it is switching to an external message distribution system, with Data Distribution Service [176] as

the default communication protocol between nodes which embeds good security and offers better portability. For these reasons ROS2 was chosen as the successor of ROS1. ROS1has its end of life scheduled for 2025 [160].

8.2.2 Existing Task Planning Frameworks based on ROS

There are three main existing solutions for task planning based on ROS: ROSPlan, Plansys2 and SkiROS2.

8.2.2.1 ROSPlan [161]

ROSPlan is the reference planning package for ROS1. It has been used for many applications for mobile robots and has seen many contributions from a big community of developers. The fact that it is ROS1 based, and that its development has slowed down with the creation of Plansys2 excluded its use for ACROBA.

8.2.2.2 Plansys2 [162]

Plansys2 is a powerful planning system framework for ROS2. It has capabilities like execution optimization by parallelization of actions and multi-robot applications with a novel auction system to affect actions to robots. The main issue encountered with Plansys2 is that the behavior tree execution is run by the Executor node. This node is dependent on the Planner node which creates the plan from the Problem expert and the domain expert. In other words, it is not possible to run a behavior tree that was designed manually. In ACROBA, the users use the graphical user interface to model their processes and then these processes are “translated” into behavior trees that communicate with ROS. As this is an important feature for the users of the ACROBA platform, plansys2 was not selected.

8.2.2.3 SkiROS2 [26]

SkiROS2 is another ROS1 based framework that offers planning based on behavior trees. They introduced the concept of extended behavior trees to optimize the time for execution of a given task. It is achieved by the parallelization of skills execution derived from a plan issued by a PDDL solver [153]. To know which skills can be parallelized, the skills are precisely described in the SkiROS2 framework (pre, hold and post conditions, parameters). The knowledge base is stored as an ontology that is edited through Protégé [173]. As SkiROS2 is ROS1 based and the support for ROS1 will end soon, SkiROS2 for ACROBA was excluded.

8.2.2.4 Navigation 2 [164]

The Navigation 2 package is an open-source project developed on the ROS2 framework. It aims to provide a safe way for mobile robots to reach their destination. The project is modular, offering the use of different planners, controllers, trajectory smoothers and most important behaviors. These behaviors are described as behavior trees.

After reviewing these possibilities, the core of the task planner was decided to be implemented on top of Navigation2 which is based on ROS2 package and offers a more flexible use of the behavior tree concept [163].

8.2.3 Behavior Tree

“A behavior tree (BT) is a way to structure the switching between different tasks in an autonomous agent, such as a robot or a virtual entity in a computer game.” [163] Behavior Trees (BTs) were originally developed in the computer game industry to improve how Non-Player Characters (NPCs) are controlled. In this industry, having modular systems is crucial because it allows developers to reuse code, build functionality step by step, and test efficiently [163]. For autonomous agents, like robots or game characters, it's important to be both reactive and modular. Reactivity means being able to quickly and effectively respond to changes - like a robot avoiding a collision when a person steps into its path, or a game character deciding whether to hide, flee, or fight when an enemy approaches [163]. Modularity, on the other hand, refers to how easily a system’s components can be broken down and recombined [163]. This allows developers to work on individual parts separately, making it easier to manage the complexity as the system grows.

BTs have become increasingly popular in the robotics industry, often replacing traditional Hierarchical Finite State Machines. The key benefits of BTs are their high modularity and reusability. Unlike state machines, BTs don’t require specific conditions to be coded for transitioning between states. Instead, the execution order is built into the tree structure itself, using control nodes that make it easy to visualize the task flow. As a result, action nodes (or "skills" in the ACROBA project) are independent of each other and can be developed separately. There are two well-known tools (Navigation2 and Groot) available for creating and managing behavior trees.

8.2.3.1 Navigation2

Navigation 2 implements the behaviortree.cpp library developed for the MOOD2Be project [165]. The modular implementation of the behavior tree engine node makes every behavior tree node a plugin. It is possible to develop custom control nodes, condition nodes, decorator nodes, and action nodes. All the action nodes implement the ROS2 action interface, which means that the behavior tree engine will call a ROS2 action server for each action node encountered. The template returns the status of the action to the behavior tree engine (RUNNING, SUCCESS OR FAILURE). The task execution is itself a ROS2 action, benefiting from the goal system to control the execution and the feedback system for monitoring it. It is possible to resend a goal to the action server to dynamically reload a behavior tree. The simple commander API (Application Programming Interface) aims at providing the navigation stack

as a library, making the use of the previously described components easier and accessible through Python 3.

8.2.3.2 Groot

Groot is a GUI developed by the creator of the behaviortree.cpp library. It is a tool to display, edit and live-play behavior trees designed and run with the behaviortree.cpp library. The communication with the live execution of the behavior tree is based on zeroMQ [167]. In ACROBA, Navigation2 is used because Groot is not supported anymore.

8.3 PROPOSED METHOD

After the literature review, the concept of the task planner was designed. At the front end it is an easy-to-use GUI based on RTMN (see Chapter 5) to plan and execute robotic processes. At the back end, the task planner is developed as an "overlay" of Navigation2, using it as a library. Other scripts and plugins are written to generate specific use cases tasks.

The task planner is part of the ACROBA platform. The main components of the ACROBA platform are the multi-modal robotic perception module and control module (comprised of the available skills of the system), the cognitive task planner module, the virtual gym or digital twin module (a simulation environment with a physics engine and visualization), the deep reinforcement learning module (used to optimize skills in the virtual gym). The task planner acts as the entry point of the platform. It is responsible for the orchestration of the other modules and the correct execution of the tasks/processes. Figure 59 provides an overview of ACROBA architecture. The details will be explained in the next paragraphs.

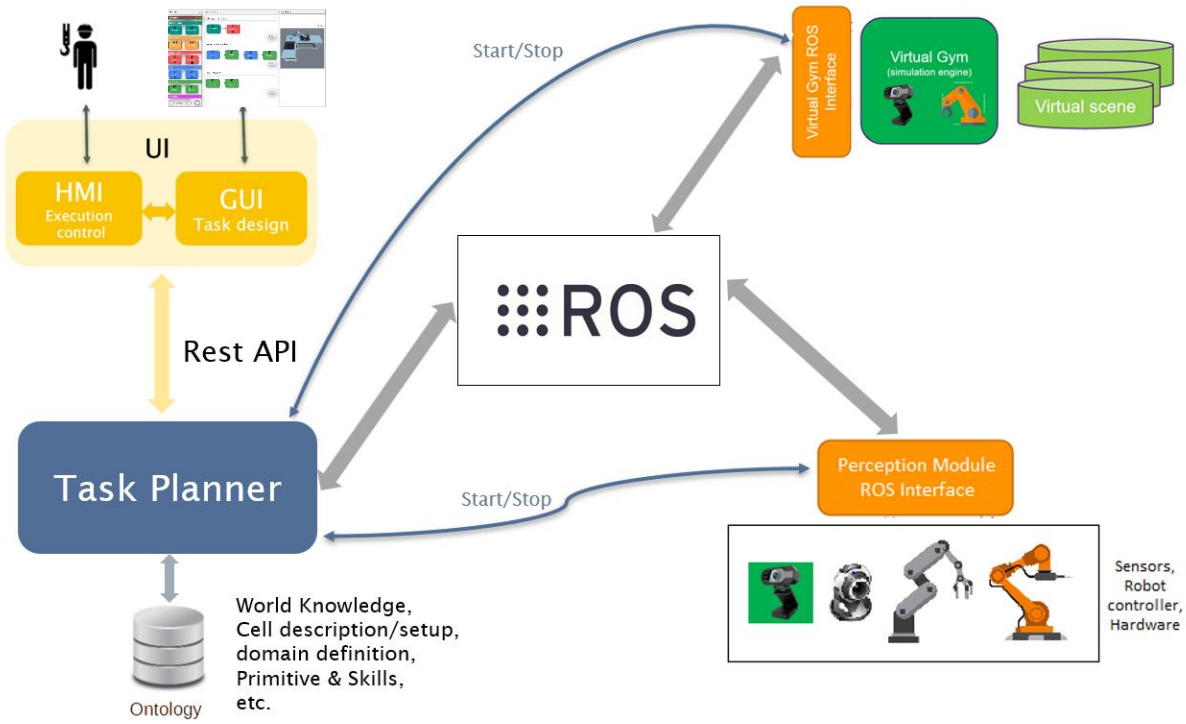


Figure 59. ACROBA Architecture Overview

8.3.1 The Task Planner

The task planner is built on ROS2. However, the rest of the ACROBA platform is developed on the ROS1 framework because ROS1 has a lot more drivers to control industrial robots and sensors. An upgrade to ROS2 is required because ROS1 is not supported anymore after 2025. Therefore, the task planner consists of ROS2, a ROS bridge to communicate with ROS1, a behavior tree writer, a GUI and an ontology.

8.3.1.1 ROS2 and the Bridge with ROS1

For the task planner to communicate with the other modules (mainly the skills package and the virtual gym), ROS bridge [168] is used. The current bridge works fine for passing messages and services between the two ROS versions. But the goal is to transmit the action communication protocol through (goal, feedback, result, status, cancel goal). The skills (real or virtual in the virtual gym) are offered as ROS1 action servers, and they are controlled by ROS2 action clients. The ROS1 action server is controlled directly by publishing and listening messages on the topics created by the server [169]. The results were satisfying at runtime while experiencing no delays or failures in the communication, but the preparation and the building processes are tedious and will, therefore, be automatized (copy of files and packages specific building order).

8.3.1.2 Behavior Tree Writer

The behavior tree writer script allows the quick creation of an XML (Extensible Markup Language) file that is read by the behavior tree engine. This method makes the writing of the behavior tree less prone error and embeds it in the task controls. Typically, it refers to the skills of the task for accessing them at runtime through service calls. Control buttons on the GUI such as pause/continue, stop/start, are also handled by the behavior tree writer.

8.3.1.3 PDDL [153]

The plan is to create PDDL Planning in the ROS2 environment that is responsible for running the Temporal Fast Downward (TFD) planner [174] for solving basic PDDL planning problems. TFD was chosen because it outperforms all state-of-the-art temporal planning systems [174]. It provides higher quality plans for a large scale of problems in comparison to plans returned by other temporal planning systems [174].

By providing the PDDL domain and problem files, the system generates a plan that is converted to a behavior tree. The planning process is considered fast for simple assembly problems that aim at optimizing the task time depending on the current state of the system. The drawback of this TFD planner is, when there are too many parts in an assembly task, the solving time gets too long. Therefore, for complex problems, the POPF (is forwards-chaining temporal planner) [175] solver is recommended as it provides a solution without considering the action duration.

8.3.2 The Interfaces

Figure 60 provides an overview of the different interfaces to the various components that interact with the task planner.

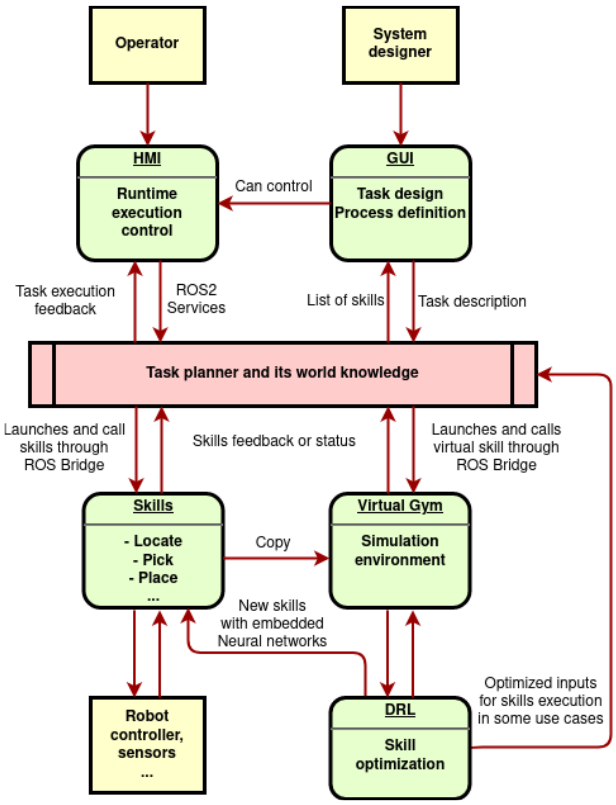


Figure 60. Task planner interfaces to other ACROBA modules

8.3.2.1 Skills

Most of the skills are implemented through ROS1 action servers. On the task planner side there are only a few skills developed to support task execution and the behavior tree engine. At run time, the task planner creates a ROS2 skills wrapper for all ROS1 skills. The ROS2 skills wrapper contains the parameters of the skills which allows the bridge to make the link with ROS1. The modular construction of the task planner with skills made of ROS2 action servers makes the calling of a skill from the task planner a direct transmission of the goals of the ROS2 server to the ROS1 server. In the same way, the ROS2 based action servers are listening to the feedback, result, and goal status of the ROS1 action server to execute the task properly. The skills used in the Virtual Gym are called virtual skills in Figure 60, just to make it clear that the skills are used in the virtual gym, they are the same as the normal skills. A simple example of the concept of the skills is shown below in Figure 61. More details about the task-skill-primitive structure are described in Chapter 7. In short, the task-skill-primitive structure enables the reuse of primitives and skills (which are robot programming units) and intuitively combines them to achieve flexible robot programming.

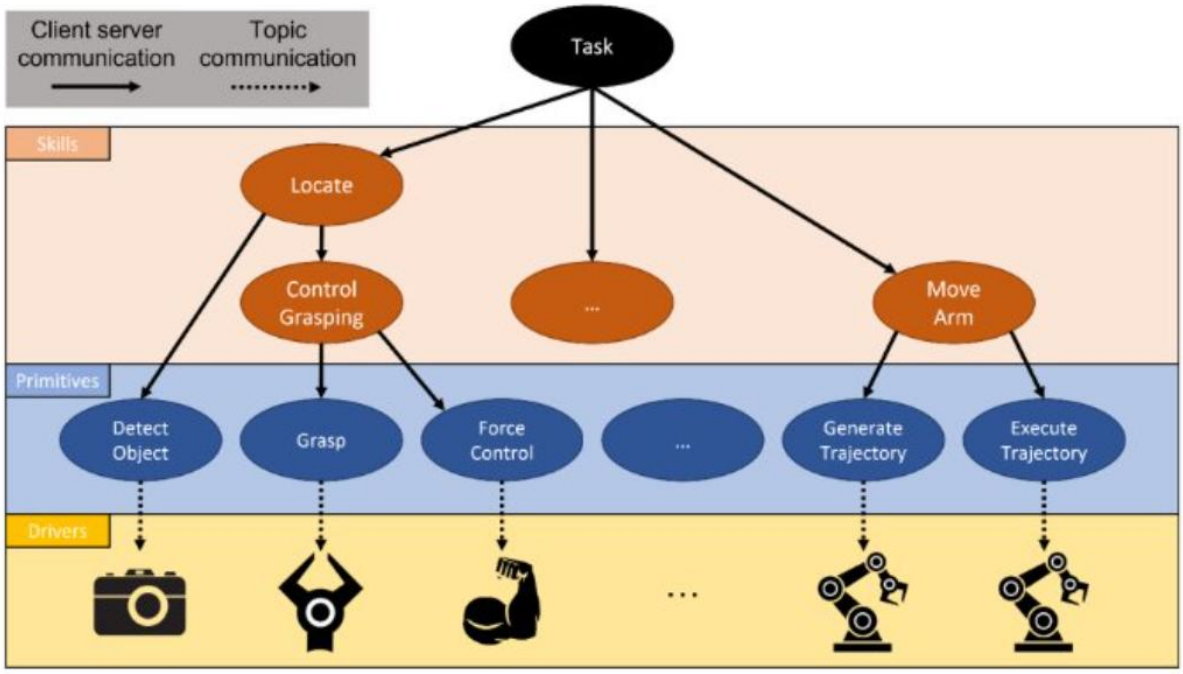


Figure 61. The Skills Concept

8.3.2.2 HMI

The task planner is meant to be running autonomously in case of a fully automated production line (without collaboration with human operators). But sometimes in the case of failure, or during the commissioning, it is important to be able to control the execution of the task (like when the operator wants to enter the cell, or the operator wants to test a skill etc...). In the case of collaborative applications, the human operator must be able to interact with the system (acknowledge the end of a manual operation, stop the robot from intervening, etc...). Therefore, we developed some HMI (Human Machine Interface) functionalities in the GUI. They are implemented as buttons:

- Start the execution
- Pause the execution
- Stop/Cancel the execution
- Restart the execution from the beginning

The HMI also aims at displaying the current execution status (the current running task/skill/primitive is highlighted) and error diagnosis (when an error occurs, a pop-up window will show the error message, once the error is solved, one can use the HMI buttons to continue).

8.3.2.3 GUI

The GUI is interfaced with the task planner in a bi-directional way. The task planner provides the GUI with its knowledge of the use case (robot used, skills available, parts to be assembled for example). The GUI provides a sequence of skills to the task planner, which generates the behavior tree.

The GUI developed for ACROBA is based on RTMN as described in Chapter 5. RTMN is a customized modelling language based on BPMN with the aim of covering the robotics domain. The reason for developing such a modelling language is based on the quantitative and qualitive research data gathered from the ACROBA users, as well as the gap identified from literature, that is the need for an intuitive graphical user interface that enables robot process modelling and execution and at the same time eases robot programming. The extensions in RTMN 2.0 (Chapter 6) are added to the GUI.

8.3.2.4 Virtual Gym and DRL Modules

The virtual Gym is used to visualize and check the execution of a task. It is launched from the task planner as a button in the GUI. Once the button is pressed, the virtual gym is opened in a new window, simulation can be managed in this window. The virtual gym can be also used to train a Deep Reinforcement Learning (DRL) module on some specific skills (for example to optimize the MoveTo skill for the generation of a better trajectory). After the training, it is saved as a new skill that will be available in the GUI for further task design.

8.3.2.5 Ontology

The ontology in Figure 65 refers to the ORPP ontology described in Chapter 7. Figure 62 shows the link between the ontology, GUI, the task planner and ROS are shown.

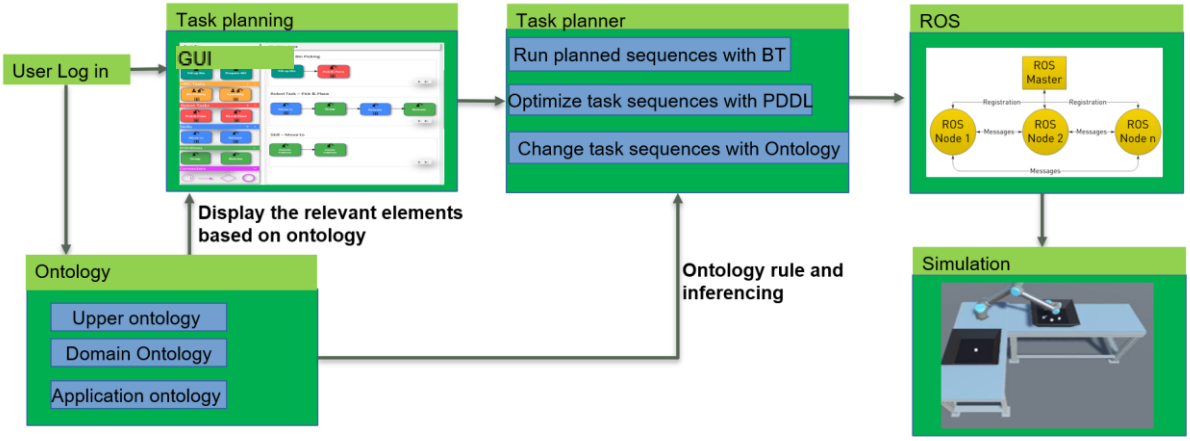


Figure 62. Task Planner Link to GUI, Ontology and ROS

The ontology has ACROBA related knowledge in the application ontology. This means once a user logs into the system the system knows this particular user belongs to an organization which in turn produces certain products. The use is linked to processes as its owner or user. Processes are composed of tasks. Tasks contain skills. Skills are made of primitives. Based on this information saved in the ontology, the system will display this user with the processes, tasks, skills, and primitives that are relevant for him/her. Then the user might choose a process to execute. He/She presses the button “start” and the order is sent to the task planner through the GUI API (Application Programming Interface). The task planner receives a sequence of tasks from the GUI and converts the sequence to a behavior tree that communicates with ROS.

8.4 FUTURE ADD-ONS

8.4.1 Assembly Sequence Planner

The aim of an Assembly Sequence Planner (ASP) is to automatically generate robotic tasks from a CAD assembly design/file. There is an increasing focus on ASP in recent research. Different methods and algorithms have been developed to try to convert a CAD assembly into a robotic task. There are also other methods to generate assembly sequences, such as integrating the assembly order information at the design phase in the CAD software [170], or analytical solving with constraints analysis [171], or an interface where the designer or the process engineer can show the order of assembly in CAD environment. According to this review [172], different algorithms and methods fit best with different types of assemblies.

Within the ACROBA project interviews (qualitative method) will be employed to decide on the best way to generate assembly sequences based on CAD data.

8.4.2 Direct Access to Primitives for Task Duration Optimization

The extended behavior tree concept (developed in Skiros2) offers a good way to optimize the task time by automatically parallelizing primitive execution when possible. The task total time is then reduced. It requires a solid knowledge of the conditions of each primitive (the primitive structure also has pre and post conditions). This feature is particularly useful when several robotic arms are working on the same process. The possibility of this add-on will be evaluated in the ACROBA project. The modular implementation of the skills on the ACROBA platform will enable this improvement as the primitives are also developed as ROS1 action servers in the same way as the skills.

8.5 CONCLUSION

In the ACROBA project, the goal of the task planner is to plan robot tasks as flexible as possible. The ROS2 base makes it viable in the future and benefits from a better communication protocol than the first version. The bridge allows the development of robotics skills in ROS1 as well as in ROS2. Building the task planner on top of Navigation2 allows the use of behavior trees for the task definition, a flexible and modular solution by design which allows the reloading of a new behavior tree at runtime. From this core, it is possible to develop scripts that automatically generate the behavior trees or a GUI where the operator can directly redesign and test the robotic task. Those scripts make use of the PDDL solving at runtime but also at the engineering phase. In the on-going development, ontologies are used to generate PDDL in an easier way. They are written from assembly design data coming directly from the factory, helping the robotic platform to quickly adapt to a new batch production.

In Chapter 9 experiments are shown to demonstrate the comprehensive use of the task planner, the RTMN and the ORPP ontology.

9 RESULTS - EXPERIMENTAL VALIDATIONS

9.1 INTRODUCTION

In this chapter, the author provides three demonstrations to validate the research which was done in the PhD study (see Figure 63). The first two demonstrations show how task planning (Chapter 8) works using the RTMN (Chapter 5) and the ORPP ontology (Chapter 7). The third demonstration shows RTMN application in human robot collaboration (Chapter 6).

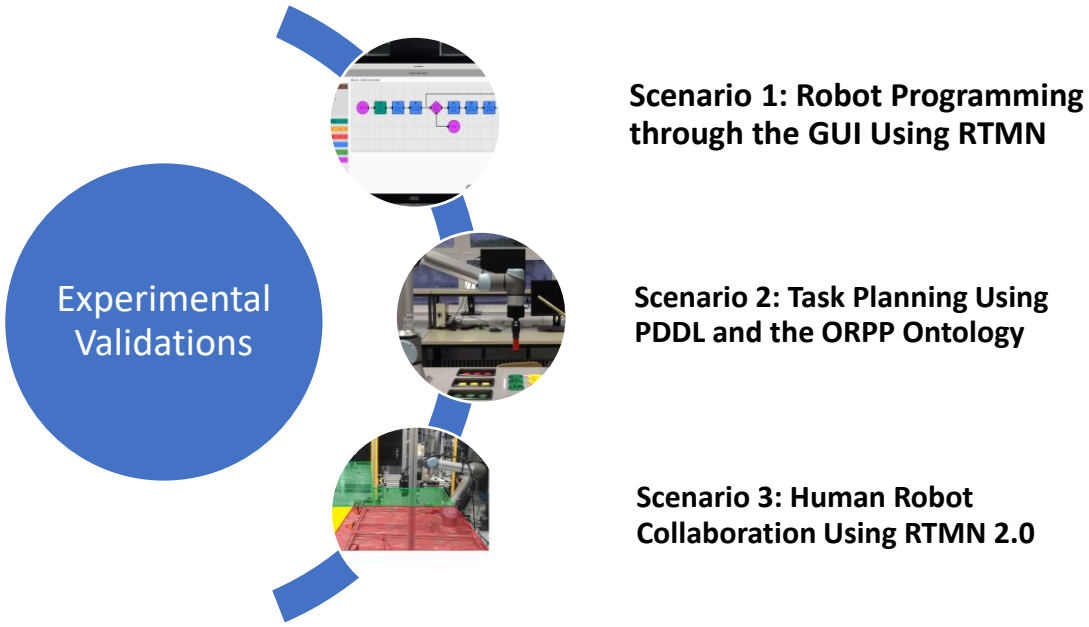


Figure 63. Experimental Validation with 3 Demonstrations

9.2 SCENARIO 1: ROBOT PROGRAMMING THROUGH THE GUI USING RTMN

This scenario is built on the technology development of RTMN in Chapter 5 and Chapter 6. It shows how a robotic process can be modelled through the graphical user interface using RTMN. The next sections describe the setup of the scenario and step-by-step modelling.

The cell setup is the same as in Chapter 7, Section 7.5.5.1 Cell Setup (see Figure 55. The Demonstrator).

The constraints, scene initial state and goal state are also the same as in Chapter 7, Section 7.5.5.2 The Scenario (see Figure 56. Initial State and Figure 57. Goal State).

9.2.1 Process/Task Description

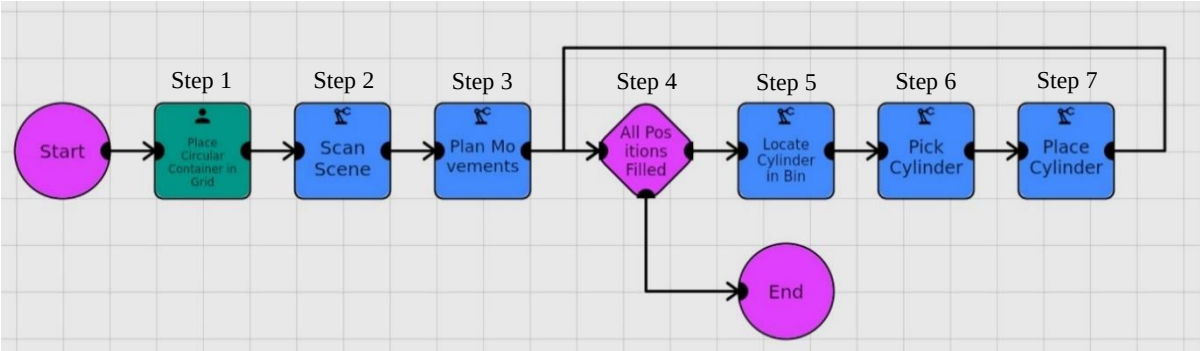


Figure 64. Assembly Process Flow

The process is presented in Figure 64. It starts with a task that is done by a human (Human Task) to place the circular-container in the grid. On the grid one can put one circular-container in each hole, in each circular-container one can put a cylinder. The operator may also put some cylinders already on the circular-containers. Then the robot scans the scene to register the grid image. Afterwards, the movement of the robot is planned. From there on the robot starts to pick the cylinder from the bin and place it at the circular-container. The color of the cylinder is given based on the color of the circular-container at position. The task is to assemble the cylinders and circular-container in the grid. The assembly goal is to put cylinders into circular-containers on all holes of the grid with matching colors.

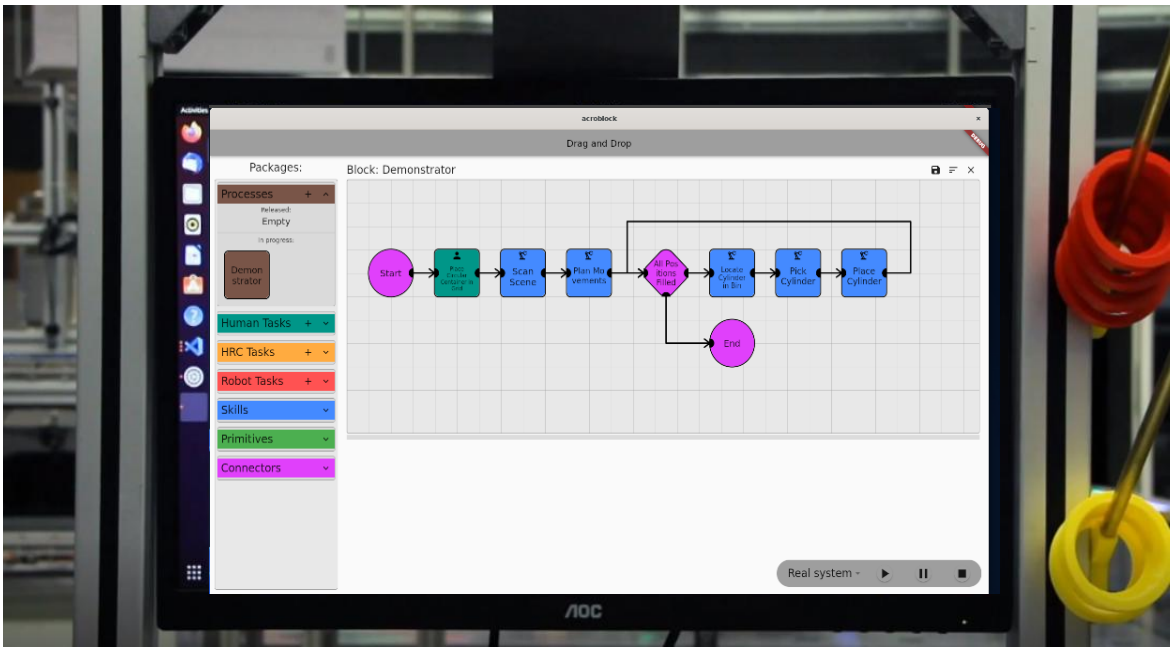


Figure 65. The Graphical User Interface

9.2.2 Modelling in the GUI

The GUI uses RTMN as the modelling language and provides the users with a graphical user interface which has RTMN notations as drag and drop elements (see Figure 65). Below the process of model is created in the GUI is described. There is one human task and six robot skills in the process. They are added sequentially.

- Step 1 (See Figure 64): Place Circular-Container in Grid

The human task – "Place Circular-Container in Grid" has the following properties (see Figure 66):

  - Name: Place Circular-Container in Grid
  - Type: Human Task
  - Responsible: Erika Muster (one of the example names in the drop-down list).

Once the process start, Erika Muster (the responsible) will receive an Email to start the "Place Circular-Container in Grid" Task in the GUI.

The "Play" Button on the main screen can be used when the task is completed. Once the button is clicked, the process continues.

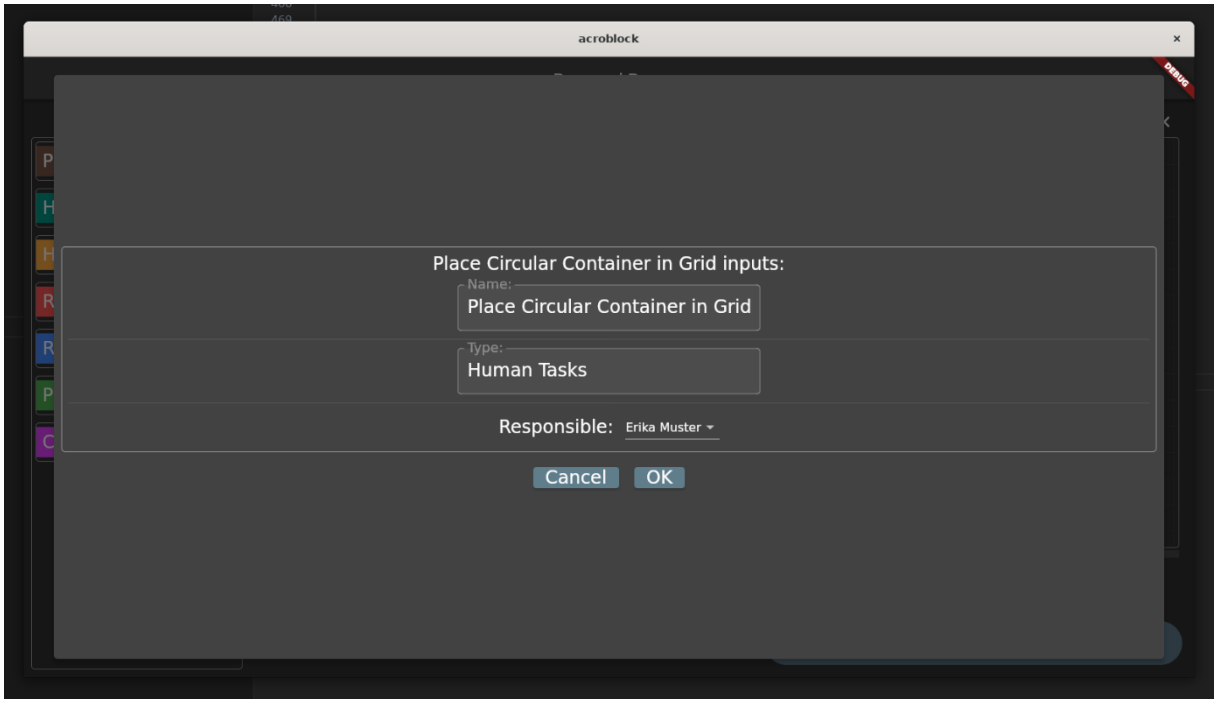


Figure 66. Place Circular-Container in Grid Property Screen

- Step 2 (See Figure 64): Scan Scene

The robot skill – "Scan Scene" has the following properties (see Figure 67):

  - Name: Scan Scene
  - Type: Robot Skill
  - Category: Vision Skill
  - Input: Image
  - Output: Scanned Picture

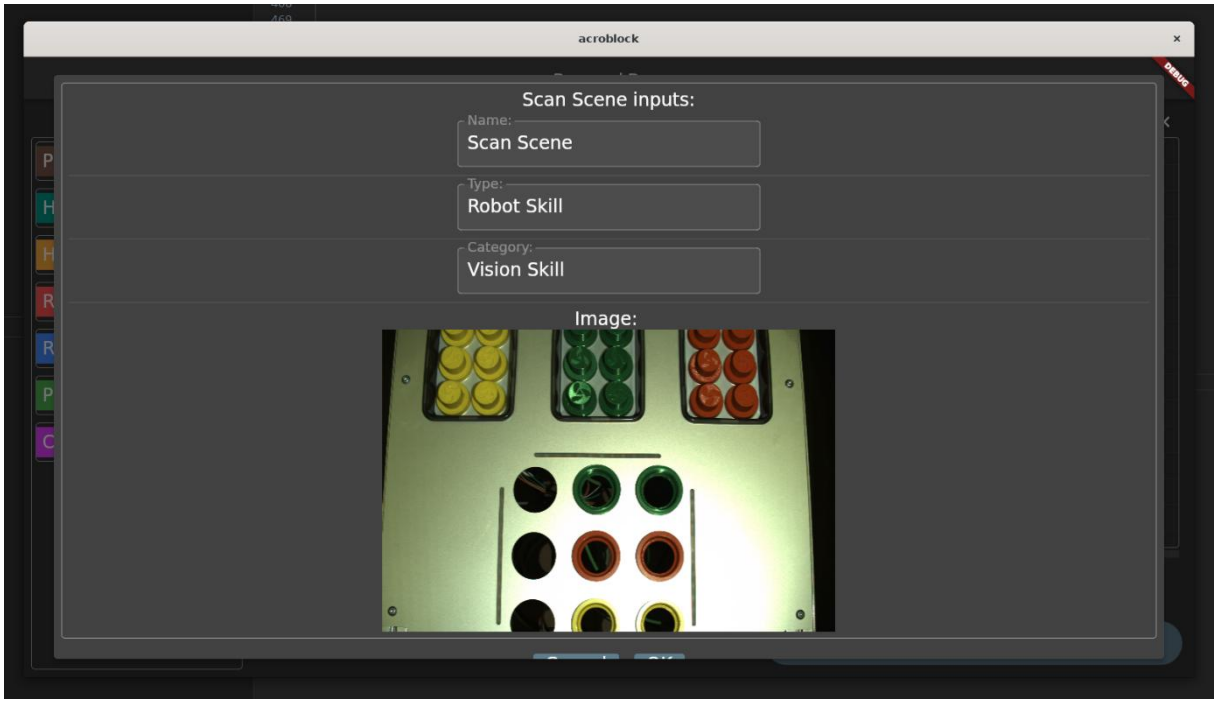


Figure 67. Scan Scene Property Screen

- Step 3 (See Figure 64): Plan Movements

The robot skill – "Plan Movements" has the following properties (see Figure 68):

- Based on the scanned scene picture, this skill calculates which position needs which color cylinder.
- Name: Plan Movements
- Type: Robot Skill
- Category: Planning Skill
- Input: Scanned Picture
- Output: Movement Position

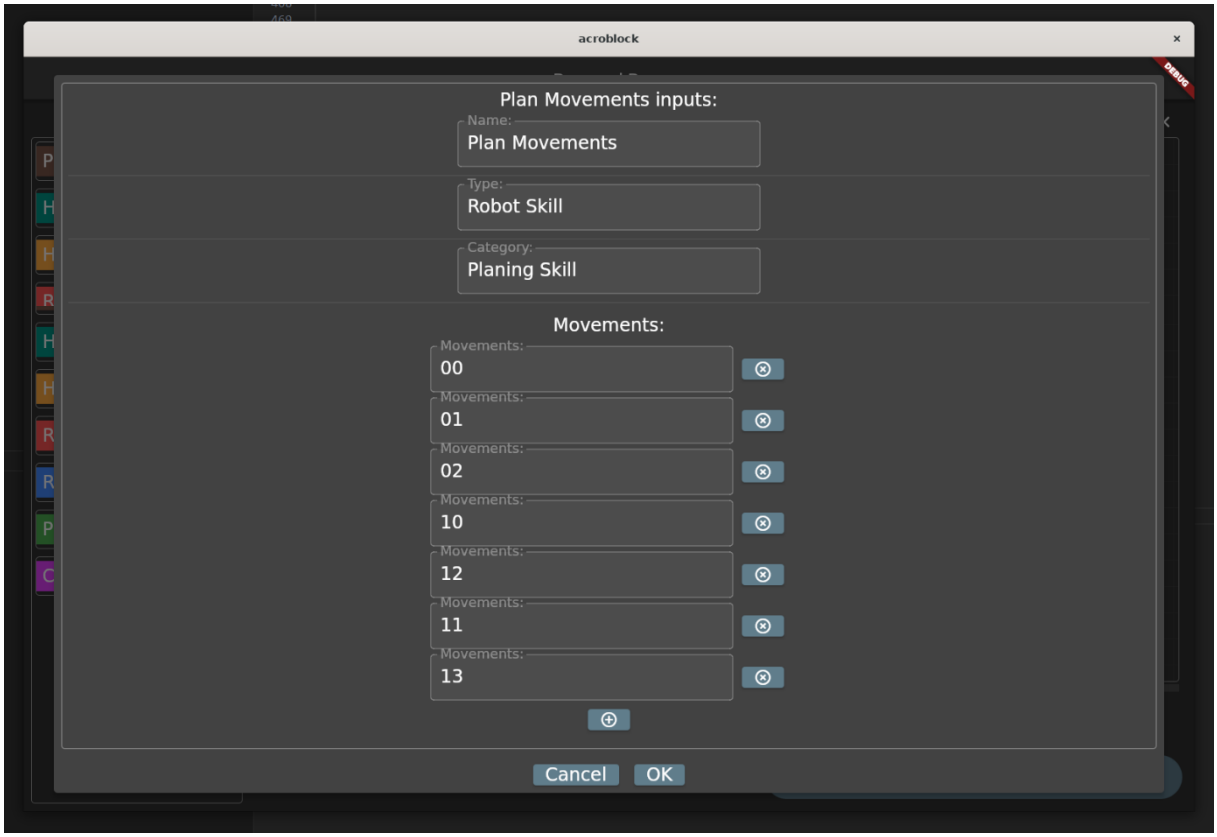


Figure 68. Plan Movements Property Screen

- Step 4: Condition – Goal reached?

The condition – "Goal reached?" has the following properties:

  - Input: Movements (00, 01, 02, 10,12,11,13), Count = 7
  - Count > 0? Yes, continue the yes branch; No, exit.
  - Output: Count = Count-1

- Step 5 (See Figure 64): Locate Cylinder in Bin

The robot skill – "Locate Cylinder in Bin" has the following properties (see Figure 69):

  - Name: Locate Cylinder in Bin
  - Type: Robot Skill
  - Category: Vision Skill
  - Input: Camera Scan Picture
  - Output: Grasping Pose

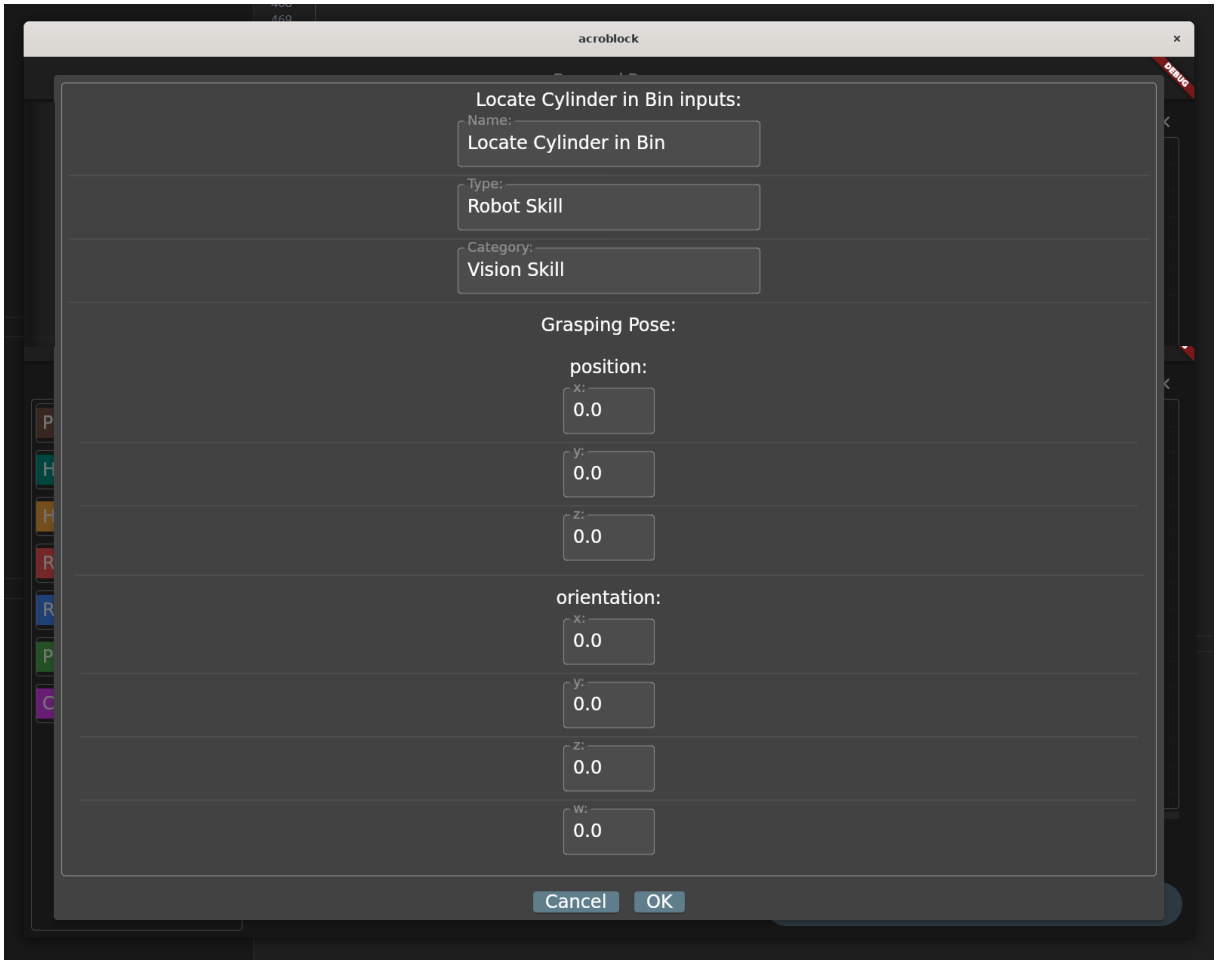


Figure 69. Locate Cylinder in Bin Property Screen

- Step 6 (See Figure 74): Pick Cylinder

The robot skill – "Pick Cylinder" has the following properties (see Figure 70):

- Name: Pick Cylinder
- Type: Robot Skill
- Category: Motion Skill
- Input: Grasping Pose, Gripper Gap
- Output: Gripper Full

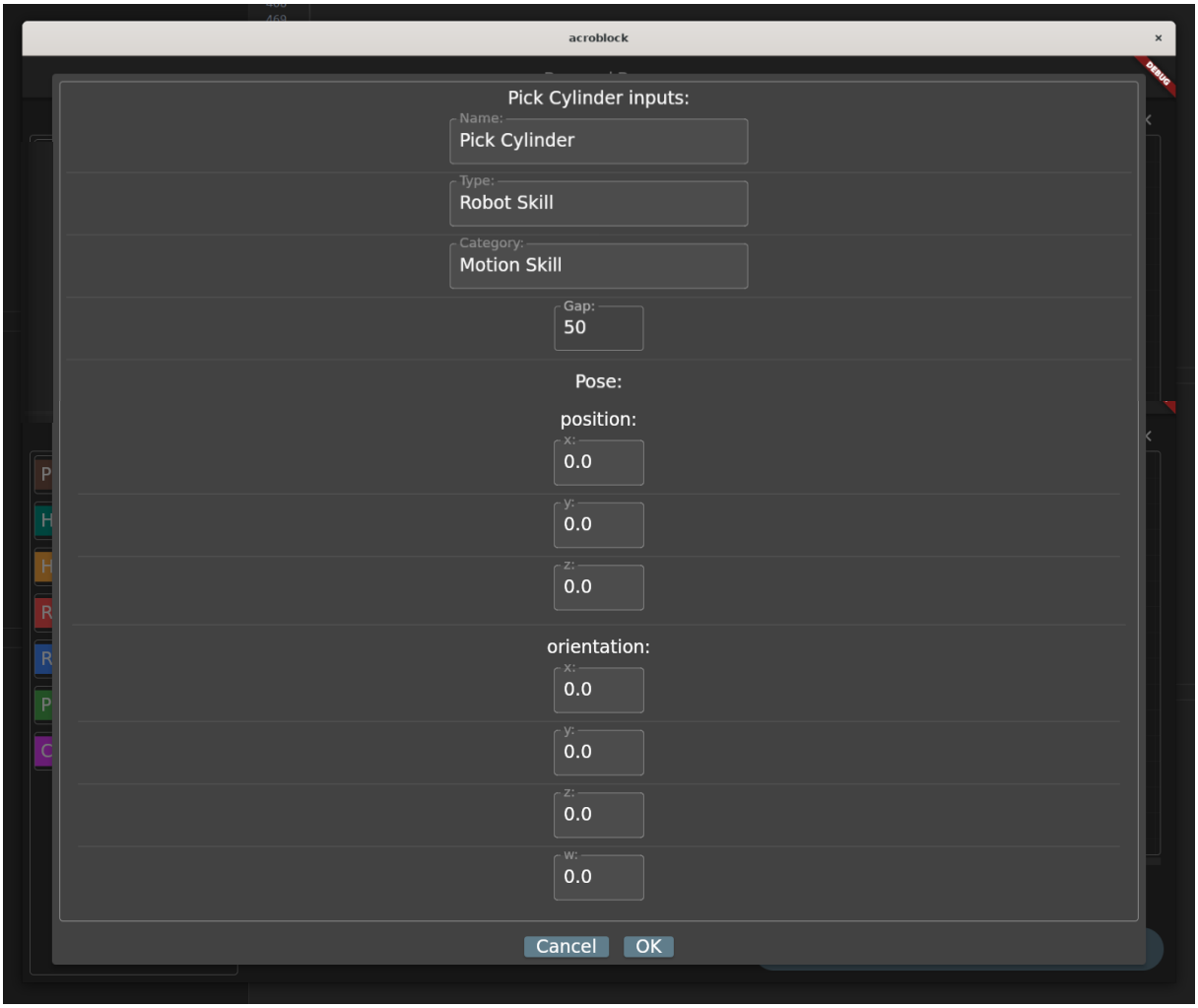


Figure 70. Plack Cylinder Property Screen

- Step 7 (See Figure 64): Place Cylinder

The robot skill – "Place Cylinder" has the following properties (see Figure 71):

- Name: Place Cylinder
- Type: Robot Skill
- Category: Motion Skill
- Input: Grasping Pose, Gripper Gap
- Output: Gripper Empty

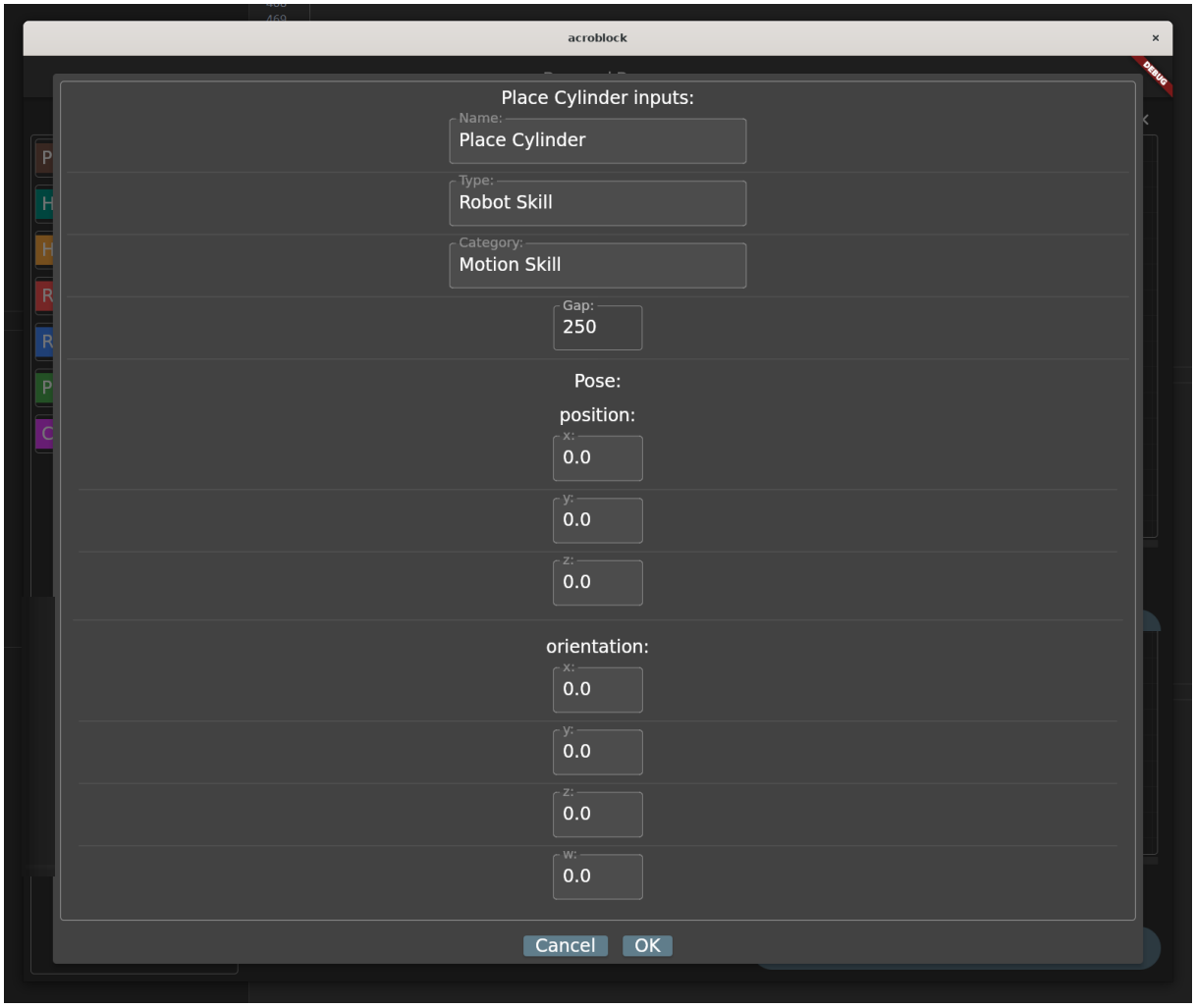


Figure 71. Place Cylinder Property Screen

9.2.3 Robot Demonstration

The robot demonstration is presented here using photos of the scenes.

Step 1: First the operator prepares the grid and puts circular-containers and cylinders in the grid based on the given initial state (see Figure 72 and 73).

Figure 72. Step1a

Figure 73. Step1b

Step 2: The robot scans the gid area and records the initial state (see Figure 74).

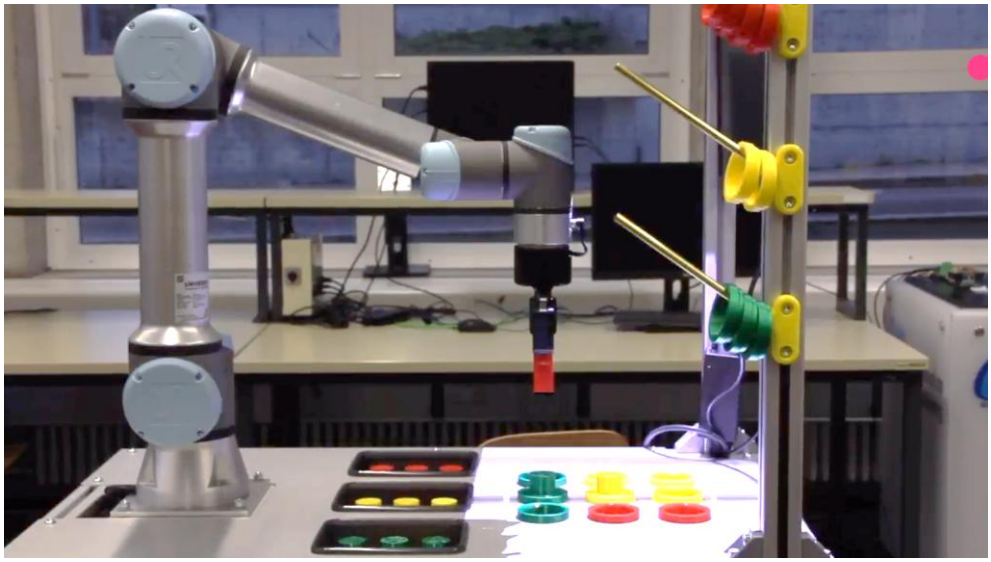


Figure 74. Step2

Step 3: The robot plans all movements in the background.

Step 4: Condition check in the background whether all positions are filled. If yes, the robot ends the task, if no, robot continues to the next step.

Step 5: The robot moves to the bin, and then locates the cylinder in the bin (see Figure 75 and 76).

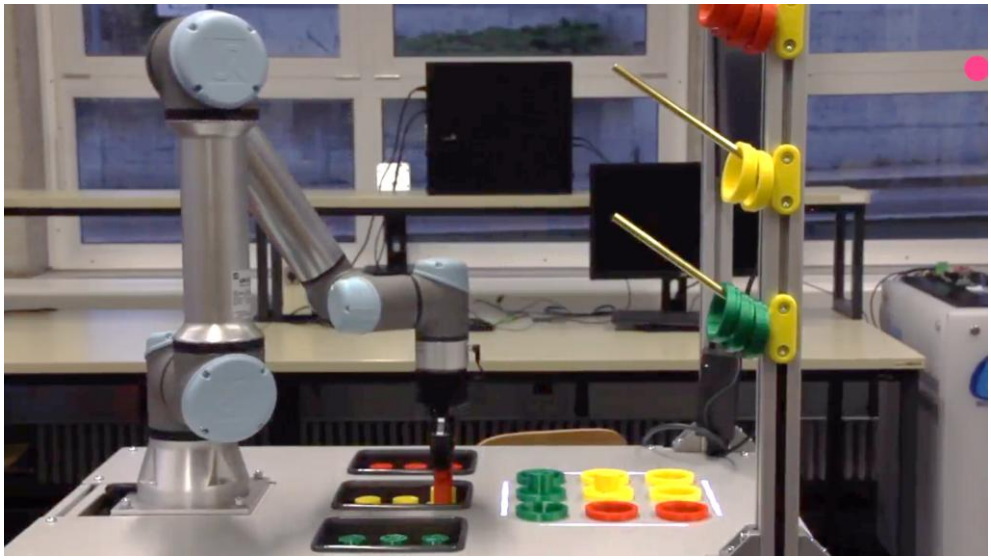


Figure 75. Step5a Move to Bin

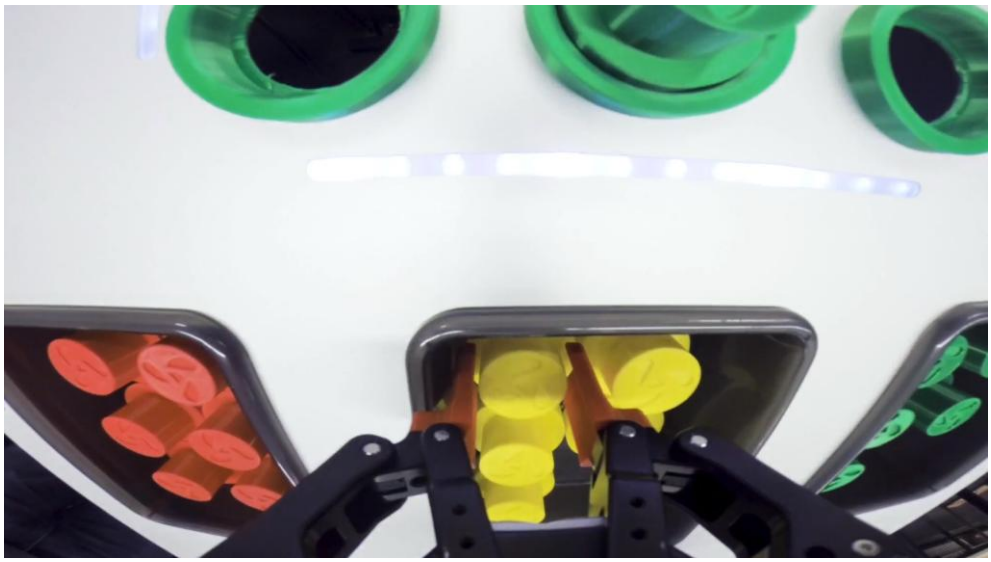


Figure 76. Step5b Locate cylinder

Step 6: The robot picks a yellow cylinder and moves to the place position.

Figure 77. Step6

Step 7: The robot places the yellow cylinder in the yellow circular-container.

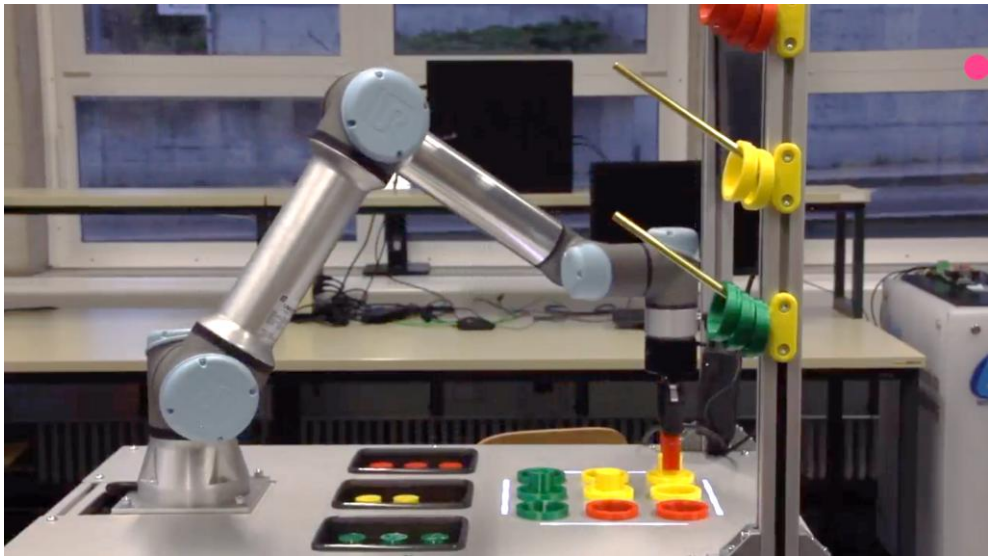


Figure 78. Step7

The robot repeats Step5 – Step7 until all circular-containers are filled (see Figure 79 and 80).

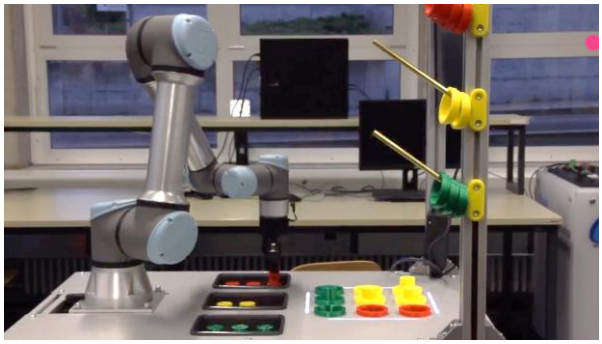


Figure 79. Pick

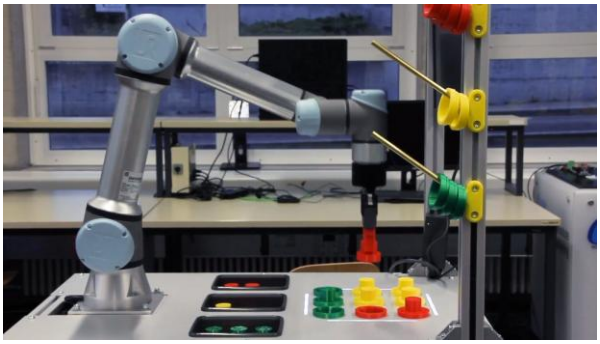


Figure 80. Place

The process is finished, and the robot returns to its base position (see Figure 81).

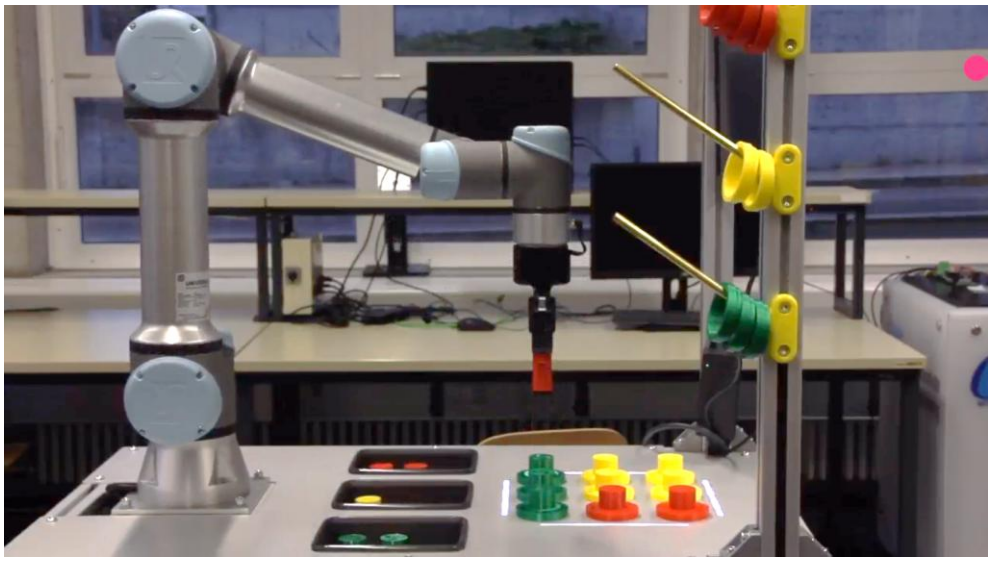


Figure 81. Task Finish

9.3 Scenario 2: Task Planning Using PDDL and the ORPP Ontology

This scenario is built on the technology development of the ORPP ontology in Chapter 7. It shows how a robotic assembly task can be defined in PDDL and in ORPP ontology as well as how the ontology can be queried to support task planning. The next sections describe the details of the scenario.

9.3.1 The Setup

The cell setup is the same as in Chapter 7, Section 7.5.5.1 Cell Setup (see Figure 52. The Demonstrator). The constraints, scene initial state and goal state are also the same as in Chapter 7, Section 7.5.5.2 The Scenario (see Figure 53. Initial State and Figure 54. Goal State). The

assembly goal is to put cylinders into circular-containers on all holes of the grid with matching colors.

The scene initial state and goal state are different from Chapter 7.

9.3.1.1 The Initial State

The user decides and sets the initial state for all positions: (00, green circular-container, no cylinder), (01, yellow circular-container, no cylinder), (02, no circular-container, no cylinder), (10, yellow circular-container, green cylinder), (11, green circular-container, no cylinder), (12, red circular-container, no cylinder), (20, green circular-container, no cylinder), (21, green circular-container, no cylinder), (22, no circular-container, no cylinder). It is presented in Figure 82.

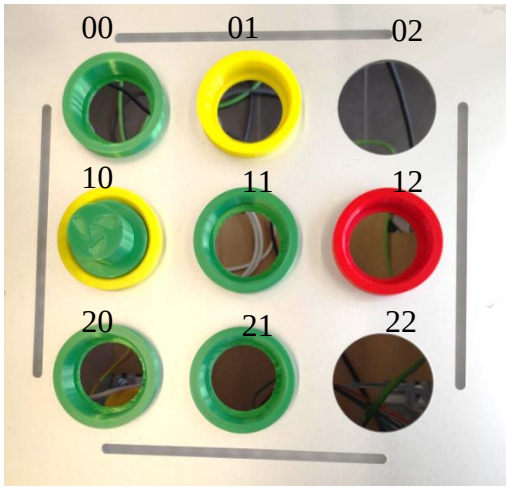


Figure 82. Scenario 2 Initial State

9.3.1.2 The Goal State

The user sets the goal state which is to put cylinders into circular-containers in all holes of the grid with matching colors.

The goal state (see Figure 83) is given for all positions: (00, red circular-container, red cylinder), (01, yellow circular-container, yellow cylinder), (02, green circular-container, green cylinder), (10, red circular-container, red cylinder), (11, yellow circular-container, yellow cylinder), (12, green circular-container, green cylinder), (20, red circular-container, red cylinder), (21, yellow circular-container, yellow cylinder), (22, green circular circular-container, green cylinder).

Figure 83. Scenario 2 Goal State

9.3.1.3 The Constraints

The robot can move one thing at a time, the cylinder can be placed on a circular-container, the circular-container cannot be put on the cylinder. Circular-containers can be put in holes; cylinders are prohibited from being put in holes.

9.3.2 PDDL (Planning Domain Definition Language) Definitions

The domain description and problem description are defined in the PDDL.

9.3.2.1 PDDL Domain Description

```
(define (domain assembly-dom)
(:types
    hole cylinder circular-container bin colour
)
(:predicates
    (hasColour ?obj - cylinder ?colour - colour)
    (contains ?bin - bin ?cylinder - cylinder)
    (empty ?bin - bin)
    (onGrid ?container - circular-container ?hole - hole)
    (placed ?cylinder - cylinder ?container - circular-container)
    (goal-state-reached)
)
(:action pick
    :parameters (?cylinder - cylinder ?bin - bin)
    :precondition (and (contains ?bin ?cylinder) (empty ?cylinder))
    :effect (and (not (contains ?bin ?cylinder)) (not (empty ?cylinder)))
)
(:action place
    :parameters (?cylinder - cylinder ?container - circular-container ?hole - hole)
    :precondition (and (empty ?container) (onGrid ?container ?hole) (not (placed ?cylinder
?container)))
    :effect (and (not (empty ?container)) (placed ?cylinder ?container)))
)
```

9.3.2.2 PDDL Problem Description

```
(define (problem assembly-prob) (:domain assembly-dom)
(:objects
    h00 h01 h02 h10 h11 h12 h20 h21 h22 - hole
    c1 c2 c3 c4 c5 c6 c7 c8 c9 c10 c11 c12 c13 c14 c15 c16 c17 c18 - cylinder
    cc00 cc01 cc02 cc10 cc11 cc12 cc20 cc21 cc22 - circular-container
    bin-red bin-yellow bin-green - bin
    red yellow green - colour
)
(:init
    ; initial state of bins
    (contains bin-red c1) (hasColour c1 red)
    (contains bin-red c2) (hasColour c2 red)
    (contains bin-red c3) (hasColour c3 red)
    (contains bin-red c4) (hasColour c4 red)
    (contains bin-red c5) (hasColour c5 red)
    (contains bin-red c6) (hasColour c6 red)
    (contains bin-yellow c7) (hasColour c7 yellow)
    (contains bin-yellow c8) (hasColour c8 yellow)
    (contains bin-yellow c9) (hasColour c9 yellow)
    (contains bin-yellow c10) (hasColour c10 yellow)
    (contains bin-yellow c11) (hasColour c11 yellow)
    (contains bin-green c13) (hasColour c13 green)
    (contains bin-green c14) (hasColour c14 green)
    (contains bin-green c15) (hasColour c15 green)
    (contains bin-green c16) (hasColour c16 green)
    (contains bin-green c17) (hasColour c17 green)
    ; Initial state of circular containers on the grid
    (onGrid cc00 h00) (hasColour cc00 green)
    (onGrid cc01 h01) (hasColour cc01 yellow)
    (onGrid cc10 h10) (hasColour cc10 yellow)
    (onGrid cc11 h11) (hasColour cc11 green)
    (onGrid cc12 h12) (hasColour cc12 red)
    (onGrid cc20 h20) (hasColour cc20 green)
    (onGrid cc21 h21) (hasColour cc21 green)

    ; Initial state of cylinders in circular containers
    (placed c13 cc10)
    ; Initial state of bins
    (not (empty bin-red)) (not (empty bin-yellow)) (not (empty bin-green))
)
(:goal (and
    (onGrid cc00 h00) (hasColour cc00 red)
    (onGrid cc01 h01) (hasColour cc01 yellow)
    (onGrid cc02 h02) (hasColour cc01 green)
    (onGrid cc10 h10) (hasColour cc10 redw)
    (onGrid cc11 h11) (hasColour cc11 yellow)
    (onGrid cc12 h12) (hasColour cc12 green)
    (onGrid cc20 h20) (hasColour cc20 red)
    (onGrid cc21 h21) (hasColour cc21 yellow)
    (onGrid cc21 h21) (hasColour cc21 green)
    (placed c1 cc00) (placed c7 cc01) (placed c13 cc02)
    (placed c2 cc10) (placed c8 cc11) (placed c14 cc12)
    (placed c3 cc20) (placed c9 cc21) (placed c15 cc22)
    (goal-state-reached)
))
)
```

9.3.3 Ontology Supporting PDDL

9.3.3.1 Ontology represented in OWL

This ontology provides a basic mapping of the PDDL concepts into an ontological format. It can support the PDDL by providing a structured and semantically rich representation of the domain concepts and relationships. Ontology (RDF/OWL) is a more general knowledge representation format, while PDDL is specifically designed for planning. To use ontology to

support the PDDL, a mapping between the ontology elements and the PDDL constructs must be established. These include:

- Types (cylinders, circular containers, bins, holes, colors, and hangers) in PDDL are represented as classes in ontology (Cylinder, Circular Container, Bin, Hole, Color, and Hanger).
- Objects are instances of types in PDDL, they are mapped to ontology with individuals. For example, individual cylinders such as c1, c2, c3 are instances of the class Cylinder, and individual circular containers such as cc1, cc2, cc3 are instances of the class CircularContainer.
- PDDL uses predicates to define relationships between objects and these are described as object properties in ontology. For example, the object property inContainer relates cylinders to circular containers, the object property inBin relates cylinders to bins, and the object property onGrid relates circular containers to holes on the grid.
- Initial state specifies the state of the world before starting in PDDL. Ontology uses object property assertions to represent these individual instances. Individual instances in the ontology are associated with their properties using object property assertions. For example, individual cylinders like c1, c2, c3 are associated with their colors and placement in bins and circular containers using object property assertions. It is the same for the goal state.
- Actions in PDDL represent possible transitions between states. They can be created as classes or properties in ontology. For example, the MoveTo action in PDDL can be defined as MoveTo Class.
- Constraints are rules and limitations for actions in PDDL. In ontology they are described using additional axioms or properties. For example, the constraint that "the container cannot be put on the cylinder" could be represented as an axiom in the ontology.
- PDDL applies reasoning to create plans, while the ontology reasoning engines can infer new information based on axioms and assertions. For example, “a container is empty” is specified in the ontology, the reasoning engine can infer and derive that the container is not holding any cylinders.

The relation graph of the ontology for PDDL domain and problem is presented in Figure 58 in Chapter 7. The ontology in OWL is presented in Appendix 15.7.

Aligning the PDDL and ontology allows for a more semantically meaningful representation of the plan domain and therefore enables more intelligent planning systems supported by ontological reasoning capabilities. More complex planning scenarios need more expressive and detailed ontology. The presented scenario does not cover complex scenarios, for instance, to allow cylinders placed in the wrong direction (top to bottom) in the initial state, or allow cylinders placed in bins randomly (on top of each other) or allow human robot collaboration.

9.3.3.2 SPARQL Queries

Ontology can be used to reason about problems, query for specific information, and facilitate problem-solving tasks such as planning and decision-making. Below a few queries one can use to get certain information from the ontology for PDDL to use are presented. The queries are defined and generated by the programmer for different purposes.

- Query to retrieve all holes and the objects currently placed on them

```
PREFIX assembly: <http://example.org/assembly#>

SELECT ?hole ?object ?color
WHERE {
  ?hole rdf:type assembly:Hole .
  OPTIONAL {
    ?object assembly:onGrid ?hole .
  }
```

These queries demonstrate different ways to retrieve information from the ontology, such as retrieving specific types of objects, checking property values, and querying relationships between objects.

The answer:

?hole	?object	?color
h00	cc00	green
h01	cc01	yellow
h02		
h10	cc10	yellow
h11	cc11	green
h12	cc12	red
h20	cc20	green
h21	cc21	green
h22		

- Query to retrieve cylinders and circular containers that are color-mismatched in the initial state

```
PREFIX assembly: <http://example.org/assembly#>

SELECT ?hole ?cylinder ?cylinderColor ?container ?containerColor
WHERE {
  ?cylinder rdf:type assembly:Cylinder ;
            assembly:hasColor ?cylinderColor ;
            assembly:inContainer ?container .

  ?container rdf:type assembly:CircularContainer ;
             assembly:hasColor ?containerColor .

  FILTER (?cylinderColor != ?containerColor)
}
```

First select the individual instances of cylinders (?cylinder) and circular containers (?container). Then retrieve the colors of cylinders (?cylinderColor) and circular containers (?containerColor) using the assembly:hasColor property. Finally, filter the results to only include instances where the color of the cylinder does not match the color of the circular container (FILTER (?cylinderColor != ?containerColor)). This query will return all cylinders and circular containers that are color mismatched in the initial state of the ontology.

The answer:

?hole	?cylinder	?cylinderColor	?container	?containerColor
h10	c13	green	cc10	yellow

9.4 SCENARIO 3: HUMAN ROBOT COLLABORATION USING RTMN 2.0

This scenario is built on the technology development of RTMN 2.0 in Human-Robot Collaboration in Chapter 6. This scenario shows how a specific setup of HRC can be implemented and used. The following paragraphs describe this scenario in detail.

9.4.1 The HRC Task/Process

Figure 84 shows the collaborative HRC task. One can consider it as a very simple process. It describes a human worker and a robot working together in parallel in the same robotic cell. The human worker enters the robot cell from time to time to check the work of the robot. It is a collapsed HRC task which contains a robot skill (Move) and a human task (Check Robot Work). Figure 85 presents the details of this task.

Figure 84. HRC Task

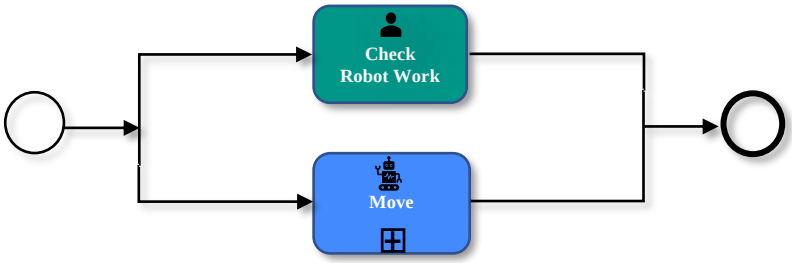


Figure 85. Components of the HRC Task

For this HRC task, the HRC type “Parallel Cooperation SSM (PS)” from Chapter 6 is used. This type combines the HRC mode 3 - Speed and Separation Monitoring and task type “parallel cooperation”. For this set-up, safety level 3(SSM) must be activated. This is achieved by using a collaborative robot, a 2D laser scanner to enable zone/position control capabilities, and 3D vision systems for monitoring speed and movement of humans.

9.4.2 Speed and Separation Monitoring

The work area is divided into three zones. These three zones ensure human safety.

- Green Zone: this is the safe zone for humans, humans can freely move in this area, robots run at full speed
- Yellow zone: this is the warning zone, when a human enters this zone the robot runs at a reduced speed. Human speed is monitored in this zone by the 3D vision system (this function is explained in Section 6.5.3 below).

- Red zone: this zone is the danger zone; the robot stops as soon as humans enter this zone.

This set up is implemented in a robotic cell IMR in Ireland where a generic collaborative cell was built for the ACROBA project. This is the only cell that is equipped with a laser scanner in ACROBA. The two collaborative use cases (IKOR&ICPE) have only light curtains in place which do not have the capability of zone separation. Below are screenshots from a testing video.

Figure 86 shows the human standing in green zone, the robot is working at full speed.

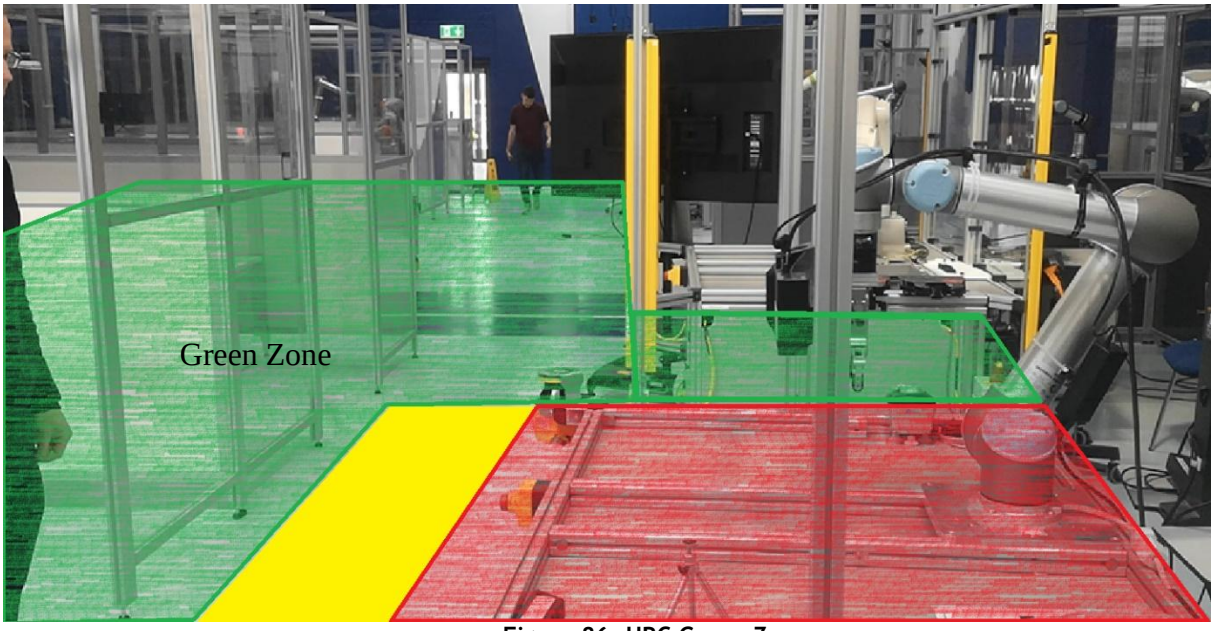


Figure 86. HRC Green Zone

Figure 87 shows the human slowly walking into the yellow zone, robot reduces its speed to collaborative speed.

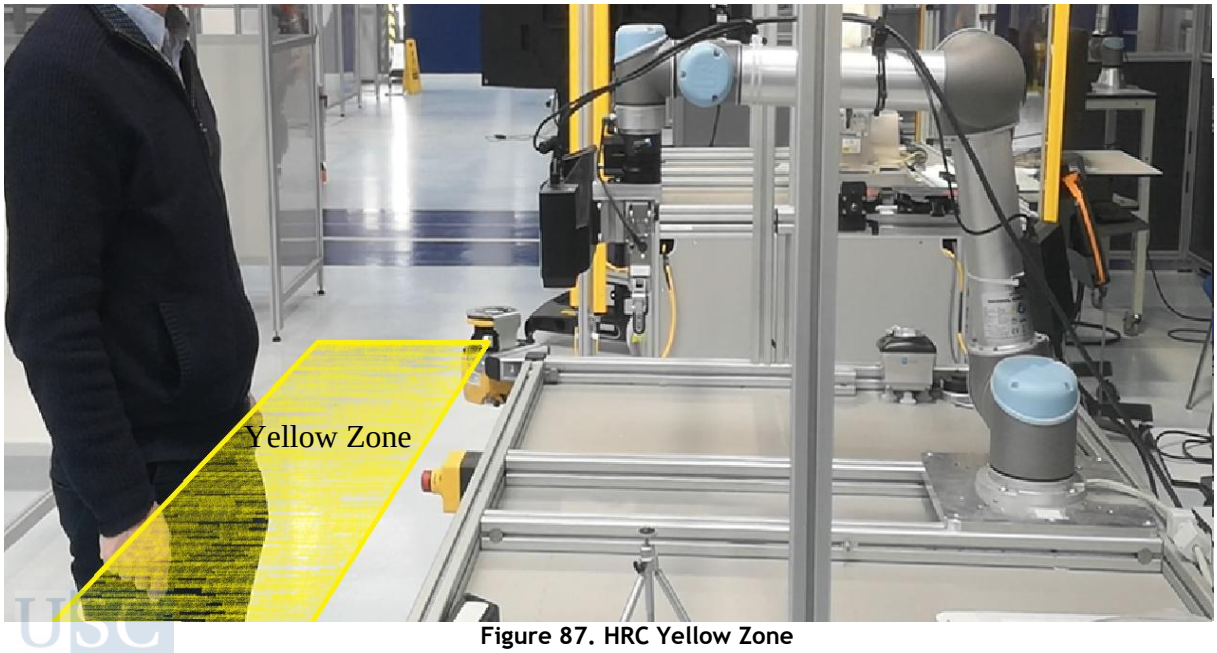


Figure 87. HRC Yellow Zone

Figure 88 shows the human approaching the red zone, the robot stops.

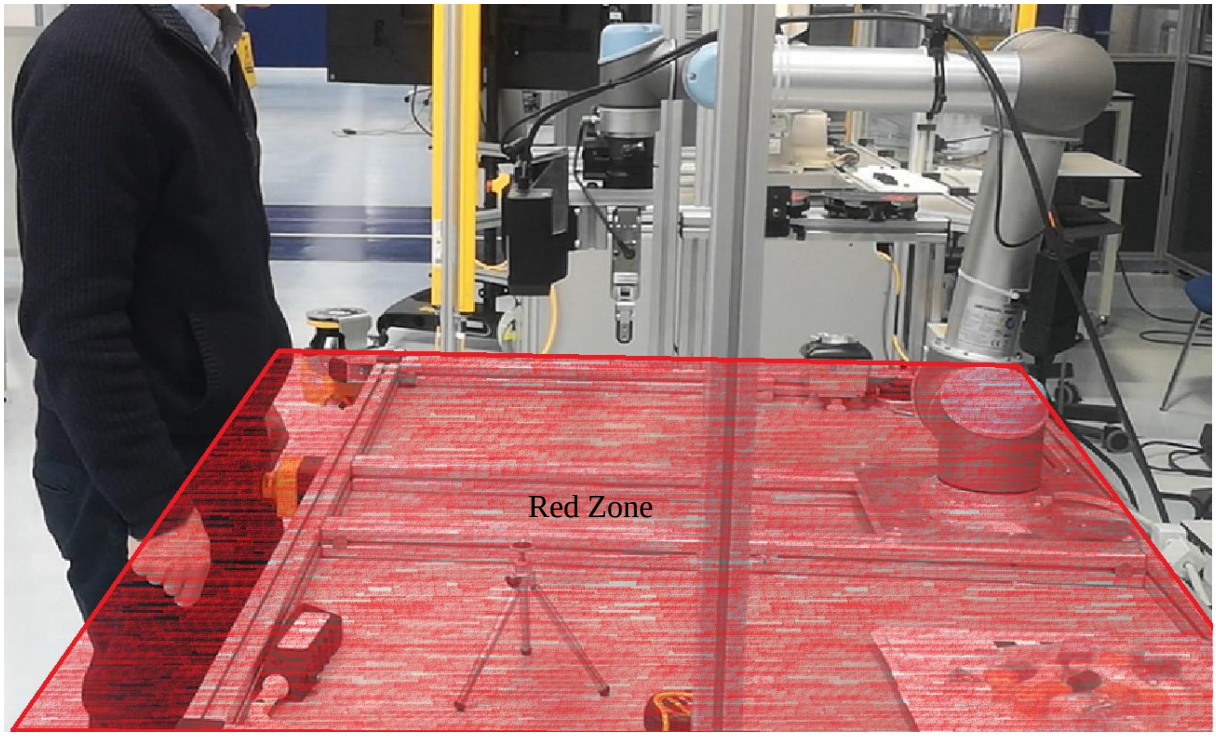


Figure 88. HRC Red Zone

9.4.3 Human Speed Monitoring

In the scenario, not only the zones are controlled, but also the speed of the human and the robot are monitored. This is programmed at the software control level. If the human is moving too quickly and a collision with robot is identified, the robot stops immediately even when the human is still in the yellow/warning zone. In other words, if the minimum separation distance between the human and the robot is reached, the robot stops.

The minimum separation distance between robot and human is calculated with the formula $D_{HR}= V_R*(T_R+T_S)+V_H*(T_R+T_S)$ described in Chapter 6. The velocity of the human is monitored by the 3D vision system using a monitoring skill – human detection. This skill is activated for this HRC task by tracking the human skeleton and speed.

From these three scenarios, the following areas of this PhD research are validated: modelling the robot process through the graphical user interface using RTMN, ontology supports task planning, and modelling HRC tasks and ensuring safety with RTMN 2.0 notations.

10 CONCLUSION

This thesis describes the process of the PhD research and the research contributions the author has made. The main goal of this research is to resolve the challenges that SMEs (Small and Medium-sized Enterprises) are facing when implementing robotic solutions in agile manufacturing. The main challenges identified are 1) to gain the rich robotic expertise required to understand the high complexity of existing robotic systems, 2) to acquire the comprehensive knowledge behind robotic systems, 3) to reduce the high programming costs for implementation. The author's main contributions are to ease these challenges by creating an intuitive modelling language to plan and control robotic processes with the support of ontology. This enables non-robotic experts to flexibly program and reprogram robots. At the same time the acquired relevant knowledge helps to achieve mass-customization.

The author has established three hypotheses and three research questions. These were answered and addressed in four publications (Chapter 5-8). The first publication (Chapter 5) was designed to confirm hypothesis 1 - there is a gap between the complex robot programming and users lacking expertise in programming and hypothesis 2 - there is a need for an easy-to-understand robot planning and controlling interface with intuitive robotic task representations, which provides a natural understanding of robotic processes to human actors). The literature review gathered the major problems operators had when using robotic systems in agile manufacturing (research question 1). These challenges were confirmed by the data collected through questionnaires and interviews. The existing modelling and programming technologies are researched to answer research question 2&3. These models and programming technologies provided the basis for the development of the modelling language RTMN (Robot Task Modelling Notation). RTMN was designed to bridge the gap between process modelling and robot execution so that even users without programming expertise are able to plan and control robots. The key concept of RTMN is to model robotic processes with task-skill-primitive structure which breaks robot processes into tasks, robot skills and skill primitives. In addition, as the basic RTMN notations are similar to the notations users use to draw their business process flows, the familiarity improves the utilization of the tool. The evaluation based on Davis's technology acceptance model [15] indicates positive future use of RTMN (validation of hypothesis 3 - An intuitive low-code process-oriented model and notation that focuses on the robotic domain will create a standardized bridge for the gap between the robotic process design and implementation.).

Although RTMN covers modelling Human Robot Collaboration tasks as a generic model, it cannot cover certain safety aspects. RTMN 2.0 (Chapter 6) - the second publication - is an extension and modification of RTMN with a focus on modelling of human robot collaboration. In addition, it modelled the requirements and KPIs to enable traceability of the business process to the lower robotic control level. Notations related to decision making are modified as an improvement to RTMN. The biggest contribution of RTMN 2.0 is the creation of synergy between HRC modes (based on ISO standards) and HRC task types in the literature. This combination generated five different HRC modelling notations: Coexistence Fence, Sequential Cooperation SMS, Teaching HG, Parallel Cooperation SSM, and Collaboration PFL. Both collaboration and safety criteria are defined for each notation. This modelling language is implemented into a graphical user interface to provide the users with the following benefits: increased intuitiveness, easy-to-use templates, it is code-free, and it ensures safety and reusability.

RTMN 2.0 eases robotic task programming by providing an intuitive interface for modelling robotic tasks. However, to achieve higher levels of intelligence for robotic programming we need the support of AI technologies. Ontology plays a significant role in AI and robotics by providing structured data, reasoning, action understanding, context awareness, knowledge transfer, and semantic learning. The structured framework created by the ontology for knowledge representation is crucial for enabling intelligent behavior in robots. For these reasons ORPP (the Ontology for Robotic Task Planning) was developed as the author's third publication (Chapter 7). ORPP aims to standardize and support robot intelligence by bridging higher-level process planning with lower-level robotic control. ORPP addresses a key gap in literature which is the need to bridge the higher-level robotic process planning and the lower-level robotic control by integrating knowledge representation and reasoning capabilities to optimize task planning in agile manufacturing. For ontology-based systems to be effective, standardization of the domain is the key. This ontology covers knowledge representation for the robotic task planning domain and provides reasoning capabilities.

RTMN 2.0 and ORPP ontology as separate applications cannot achieve holistic robotic task planning. They can only achieve their full capability when they are integrated together to support task planning. The last publication (Chapter 8) presents a comprehensive approach to combine the modelling language RTMN 2.0 and the ORPP ontology for robotic task planning. RTMN 2.0 and ORPP together aim to support and accelerate intuitive and flexible robot task planning from the front and background. The proposed task planner in this publication has a built-in graphical user interface which adopts the RTMN modelling principles. In the background, the ORPP ontology serves as the knowledge base and supports robot intelligence by querying and reasoning with the knowledge base. This approach of combining task planning, RTMN and ORPP together maximize the task planning capabilities to enable robotic task planning as flexibly as possible.

The experimental validation (Chapter 9) shown in this thesis presents how the research contributions work together and provides an overview of the robot's actions. Three demonstrations are presented to validate the research the author has done in her PhD study. The first demonstration uses RTMN modelling language based graphical use interface to model a robotic process. The second demonstration shows how task planning works using the RTMN (GUI) and the ORPP ontology. The third demonstration illustrates a human robot collaboration scenario using the modelling notation developed in RTMN 2.0.

The research goals in this PhD study have been reached. Nevertheless, further research can be carried out to further develop additional functionalities. RTMN is intuitive and flexible for modelling simple to medium complex processes, however, for complex processes that have, for example, multi-loops, RTMN lacks functionalities/modelling elements to simplify the process modelling. Further extensions should focus on adding these elements to the modelling notations. ORPP established a solid foundation for robotic task planning ontology and can be used to extend to other domain ontologies. Deepening human-robot collaboration, enhancing knowledge related to human safety and factors, expanding PDDL planning to generic and more complex scenarios are future research areas. In addition, linking/aligning to the Robot Operating System (ROS) will make ORPP ontology more adaptable and powerful (ROS is a set of software libraries and tools that help you build robot applications [16]). Therefore, further development and improvement should focus on these abovementioned areas.

11 BIBLIOGRAPHY

[1] "Project - Acroba Project." https://acrobaproject.eu/project-acroba/ (accessed Feb. 28, 2022).

[2] Y. A. Zagorulko, E. A. Sidorova, I. R. Akhmadeeva, W. Pang, W. Gu, and H. Li, "Ontology-based task planning for autonomous unmanned system: framework and principle," J Phys Conf Ser, vol. 2253, p. 12018, 2022, doi: 10.1088/1742-6596/2253/1/012018.

[3] D. I. Levine, "Automation as Part of the Solution:," https://doi.org/10.1177/1056492619827375, vol. 28, no. 3, pp. 316–318, Feb. 2019, doi: 10.1177/1056492619827375.

[4] C. Schlenoff et al., "An IEEE standard Ontology for Robotics and Automation," IEEE International Conference on Intelligent Robots and Systems, pp. 1337–1342, 2012, doi: 10.1109/IROS.2012.6385518.

[5] A. Olivares-Alarcos et al., "A review and comparison of ontology-based approaches to robot autonomy," Knowl Eng Rev, vol. 34, 2019, doi: 10.1017/S0269888919000237.

[6] S. Manzoor et al., "Ontology-Based Knowledge Representation in Robotic Systems: A Survey Oriented toward Applications," Applied Sciences 2021, Vol. 11, Page 4324, vol. 11, no. 10, p. 4324, May 2021, doi: 10.3390/APP11104324.

[7] A. Umbrico, A. Orlandini, and A. Cesta, "An ontology for human-robot collaboration," in Procedia CIRP, 2020, vol. 93, pp. 1097–1102. doi: 10.1016/j.procir.2020.04.045.

[8] M. Beetz, D. Bessler, A. Haidu, M. Pomarlan, A. K. Bozcuoglu, and G. Bartels, "Know Rob 2.0 - A 2nd Generation Knowledge Processing Framework for Cognition-Enabled Robotic Agents," in Proceedings - IEEE International Conference on Robotics and Automation, Sep. 2018, pp. 512–519. doi: 10.1109/ICRA.2018.8460964.

[9] W. Pang, W. Gu, and H. Li, "Ontology-based task planning for autonomous unmanned system: framework and principle," in Journal of Physics: Conference Series, Apr. 2022, vol. 2253, no. 1. doi: 10.1088/1742-6596/2253/1/012018.

[10] Weser M, Bock J, Schmitt S, Perzylo A, and Evers K, An Ontology-Based Metamodel for Capability Descriptions. 2020.

[11] J. W. Creswell and V. L. Plano Clahk, DESIGNING AND CONDUCTING MIXED METHODS RESEARCH, 2nd ed. SAGE, 2011.

[12] J. F. Molina-Azorin, J.-J. Tari, M. D. Lopez-Gamero, J. Pereira-Moliner, and E. M. Pertusa-Ortega, "The Implementation and Advantages of Mixed Methods in Competitive Strategy and Management Systems," Daft & Lewin, vol. 10, no. 1, pp. 412–421, 2018, doi: 10.29034/ijmra.v10n1a28.

[13] H. Norreklit and F. H. Selto, "Qualitative Research in Accounting & Management ·," 2011, doi: 10.1108/11766091111124702.

[14] U. Kelle, "Combining qualitative and quantitative methods in research practice: Purposes and advantages," Qual Res Psychol, vol. 3, no. 4, pp. 293–311, 2006, doi: 10.1177/1478088706070839.

[15] F. D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," MIS Q, vol. 13, no. 3, pp. 319–339, 1989, doi: 10.2307/249008.

[16] ROS: Home. Available online: https://www.ros.org/ (accessed on 28 February 2024).

[17] K. Bourr, F. Corradini, S. Pettinari, B. Re, L. Rossi, and F. Tiezzi, "Disciplined use of BPMN for mission modelling of Multi-Robot Systems," Proceedings http://ceur-ws. org ISSN, vol. 1613, p. 0073, 2021.

[18] D. P. Losey, C. G. McDonald, E. Battaglia, and M. K. O'Malley, "A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction," Applied Mechanics Reviews, vol. 70, no. 1, Jan. 2018, doi: 10.1115/1.4039145/443697.

[19] A. Ahmad and M. Idrees, "BBVPL: A Block-Based Visual Programming Language Built on Google's Blockly Disaster Trail Management View project," Article in International Journal of Advanced Trends in Computer Science and Engineering, 2021, doi: 10.30534/ijatcse/2021/1441032021.

[20] C. Lewis and J. Rieman, "Task-centered user interface design," A practical introduction, 1993.

[21] V. Villani, L. Sabattini, J. N. Czerniak, A. Mertens, and C. Fantuzzi, "MATE robots simplifying my work: The benefits and socioethical implications," IEEE Robotics and Automation Magazine, vol. 25, no. 1, pp. 37–45, Mar. 2018, doi: 10.1109/MRA.2017.2781308.

[22] A. G. Billard, S. Calinon, and R. Dillmann, "Learning from Humans," Springer Handbook of Robotics, pp. 1995–2014, Jan. 2016, doi: 10.1007/978-3-319-32552-1_74.

[23] V. Villani, F. Pini, F. Leali, and C. Secchi, “Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications,” Mechatronics, vol. 55, pp. 248–266, Nov. 2018, doi: 10.1016/J.MECHATRONICS.2018.02.009.

[24] M. Winterer, C. Salomon, J. Köberle, R. Ramler, and M. Schittengruber, An Expert Review on the Applicability of Blockly for Industrial Robot Programming; An Expert Review on the Applicability of Blockly for Industrial Robot Programming, vol. 1. 2020. [Online]. Available: http://orcid.org/0000-0002-3894-4635http://orcid.org/0000-0002-5665-2919http://orcid.org/0000-0001-9903-6107

[25] M. Colledanchise and P. Ögren, Behavior Trees in Robotics and AI. CRC Press, 2018. doi: 10.1201/9780429489105.

[26] F. Rovida, B. Grossmann, and V. Kruger, “Extended behavior trees for quick definition of flexible robotic tasks,” in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sep. 2017, pp. 6793–6800. doi: 10.1109/IROS.2017.8206598.

[27] K. Winter, I. J. Hayes, and R. Colvin, “Integrating Requirements: The Behavior Tree Philosophy,” 2010 8th IEEE International Conference on Software Engineering and Formal Methods, 2010, doi: 10.1109/SEFM.2010.13.

[28] T. Fong, C. Thorpe, and C. Baur, “Collaboration, Dialogue, Human-Robot Interaction,” Robotics Research, pp. 255–266, Aug. 2003, doi: 10.1007/3-540-36460-9_17.

[29] M. Aspridou, “Extending BPMN for modelling manufacturing processes,” Eindhoven, 2017.

[30] J.-P. de la Croix and G. Lim, “Event-Driven Modelling and Execution of Robotic Activities and Contingencies in the Europa Lander Mission Concept Using BPMN,” VIRTUAL The International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS) 2020, Pasadena, California, October 19-23, 2020. 2020. Accessed: Feb. 21, 2022. [Online]. Available: http://hdl.handle.net/2014/53293

[31] P. Bocciarelli, A. D’Ambrogio, A. Giglio, and E. Paglia, “A BPMN extension for modelling Cyber-Physical-Production-Systems in the context of Industry 4.0,” Proceedings of the 2017 IEEE 14th International Conference on Networking, Sensing and Control, ICNSC 2017, pp. 599–604, Aug. 2017, doi: 10.1109/ICNSC.2017.8000159.

[32] M. Diab et al., “FailRecOnt-An Ontology-Based Framework for Failure Interpretation and Recovery in Planning and Execution,” 2021. [Online]. Available: http://ceur-ws.org

[33] E. Verband der Elektrotechnik and Institute of Electrical and Electronics Engineers., Modularization of Skill Ontologies for Industrial Robots.

[34] E. Aguado, R. Sanz, and C. Rossi, “Ontologies for run-time self-adaptation of mobile robotic systems ,” 2021, Accessed: Sep. 05, 2022. [Online]. Available: https://www.researchgate.net/publication/355544692

[35] T. R. Gruber, “Toward principles for the design of ontologies used for knowledge sharing?,” Int J Hum Comput Stud, vol. 43, no. 5–6, pp. 907–928, Nov. 1995, doi: 10.1006/IJHC.1995.1081.

[36] N. Guarino, D. Oberle, and S. Staab, “What Is an Ontology?,” Handbook on Ontologies, pp. 1–17, 2009, doi: 10.1007/978-3-540-92673-3_0.

[37] R. Studer, V. R. Benjamins, and D. Fensel, “Knowledge engineering: Principles and methods,” Data Knowl Eng, vol. 25, no. 1–2, pp. 161–197, Mar. 1998, doi: 10.1016/S0169-023X(97)00056-6.

[38] J. Ludger and S. Schulz, “The ten commandments of ontological engineering,” in the 3rd Workshop of Ontologies in Biomedicine and Life Sciences, Oct. 2011. doi: 10.1186/2041-1480-3-S2-S1.

[39] A. Pease, I. Niles, and J. Li, “The Suggested Upper Merged Ontology: A Large Ontology for the Semantic Web and its Applications,” 2002, Accessed: Jan. 25, 2023. [Online]. Available: www.aaai.org

[40] S. Borgo et al., “DOLCE: A Descriptive Ontology for Linguistic and Cognitive Engineering *,” Appl Ontol, vol. 3, pp. 1–3, 2006, Accessed: Jan. 25, 2023. [Online]. Available: http://www.loa.istc.cnr.it/index.php/dolce/

[41] P. J. S. Goncalves et al., “IEEE Standard for Autonomous Robotics Ontology [Standards],” IEEE Robot Autom Mag, vol. 28, no. 3, pp. 171–173, Sep. 2021, doi: 10.1109/MRA.2021.3095993.

[42] “Theoretical Foundations of Ontologies,” Ontological Engineering, pp. 1–45, Apr. 2004, doi: 10.1007/1-85233-840-7_1.

[43] B. McBride, “The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS,” Handbook on Ontologies, pp. 51–65, 2004, doi: 10.1007/978-3-540-24750-0_3.

[44] G. Booch, I. Jacobson, and J. Rumbaugh, “Unified Modelling Language for Object-Oriented Development Documentation Set Version 0.91Addendum UML Update,” vol. 1, p. 14, 1996, Accessed: Jan. 25, 2023. [Online]. Available: http://www.rational.com

[45] “BPMN Specification - Business Process Model and Notation.” https://www.bpmn.org/ (accessed Feb. 27, 2022).

[46] U. Kelle, “Combining qualitative and quantitative methods in research practice: Purposes and advantages,” Qual Res Psychol, vol. 3, no. 4, pp. 293–311, 2006, doi: 10.1177/1478088706070839.

[47] “Basic Mixed Methods Research Designs - Harvard Catalyst.” Accessed: Feb. 28, 2024. [Online]. Available: https://catalyst.harvard.edu/community-engagement/mmr/hcat_mmr_sm-6090567e0f943-60905896c80af-60e5fdbc2399e-60e5fdd8057fc-610bf777da6a0-610bf7808de24-610bf792228a4-610bf8685d8f5-610bf871cbea9/

[48] “A FUTURE THAT WORKS: AUTOMATION, EMPLOYMENT, AND PRODUCTIVITY”, Accessed: Feb. 26, 2022. [Online]. Available: www.mckinsey.com/mgi.

[49] D. Norman, The design of everyday things: Revised and expanded edition. Basic books, 2013.

[50] F. D. Davis, “User acceptance of information technology: system characteristics, user perceptions and behavioral impacts,” Int J Man Mach Stud, vol. 38, no. 3, pp. 475–487, Mar. 1993, doi: 10.1006/IMMS.1993.1022.

[51] F. Abdullah and R. Ward, “Developing a General Extended Technology Acceptance Model for E-Learning (GETAMEL) by analysing commonly used external factors,” Comput Human Behav, vol. 56, pp. 238–256, Mar. 2016, doi: 10.1016/J.CHB.2015.11.036.

[52] V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis, “User acceptance of information technology: Toward a unified view,” MIS Q, vol. 27, no. 3, pp. 425–478, 2003, doi: 10.2307/30036540.

[53] R. J. K. Jacob et al., “Reality-Based Interaction: A Framework for Post-WIMP Interfaces,” Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI ’08, 2008, doi: 10.1145/1357054.

[54] Great Britain. Health and Safety Executive., “The explosion and fires at the Texaco refinery, Milford Haven, 24 July 1994 : a report of the investigation by the Health and Safety Executive into the explosion and fires on the Pembroke Cracking Company Plant at the Texaco Refinery, Milford Haven on 24 July 1994.,” p. 66, 1997.

[55] F. Nachreiner, P. Nickel, and I. Meyer, “Human factors in process control systems: The design of human–machine interfaces,” Saf Sci, vol. 44, no. 1, pp. 5–26, Jan. 2006, doi: 10.1016/J.SSCI.2005.09.003.

[56] Y. Shen, G. Reinhart, and M. M. Tseng, “A design approach for incorporating task coordination for human-robot-coexistence within assembly systems,” 9th Annual IEEE International Systems Conference, SysCon 2015 - Proceedings, pp. 426–431, Jun. 2015, doi: 10.1109/SYSCON.2015.7116788.

[57] J. Wajcman, “New connections: social studies of science and technology and studies of work:,” http://dx.doi.org/10.1177/0950017006069814, vol. 20, no. 4, pp. 773–786, Jun. 2016, doi: 10.1177/0950017006069814.

[58] A. Ahmad and M. Idrees, “BBVPL: A Block-Based Visual Programming Language Built on Google’s Blockly Disaster Trail Management View project,” Article in International Journal of Advanced Trends in Computer Science and Engineering, 2021, doi: 10.30534/ijatcse/2021/1441032021.

[59] E. Tilley and J. Gray, “Dronely: A Visual Block Programming Language for the Control of Drones,” 2017, doi: 10.1145/3077286.3077307.

[60] N. Yatapanage, K. Winter, and S. Zafar, “Slicing Behavior Tree Models for Verification,” IFIP Adv Inf Commun Technol, vol. 323 AICT, pp. 125–139, 2010, doi: 10.1007/978-3-642-15240-5_10.

[61] H. Völzer, “An Overview of BPMN 2.0 and Its Potential Use,” Lecture Notes in Business Information Processing, vol. 67 LNBIP, pp. 14–15, Oct. 2010, doi: 10.1007/978-3-642-16298-5_3.

[62] A. R. Sadik, C. Goerick, and M. Muehlig, “Modelling and Simulation of a Multi-Robot System Architecture,” Proceedings of the 2019 International Conference on Mechatronics, Robotics and Systems Engineering, MoRSE 2019, pp. 8–14, Dec. 2019, doi: 10.1109/MORSE48060.2019.8998662.

[63] J. Erasmus, I. Vanderfeesten, K. Traganos, R. Keulen, and P. Grefen, “The HORSE Project: The Application of Business Process Management for Flexibility in Smart Manufacturing”, doi: 10.3390/app10124145.

[64] R. Lindorfer and R. Froschauer, “Towards user-oriented programming of skill-based Automation Systems using a domain-specific Meta-Modelling Approach,” 2019.

[65] W. Liu, W. Zhang, B. Dutta, Z. Wu, and M. Goh, “Digital Twinning for Productivity Improvement Opportunities with Robotic Process Automation: Case of Greenfield Hospital,” 2020, doi: 10.18178/ijmerr.9.2.258-263.

[66] B. Sourabh Shivaji, J. Afaqahmed Mushtaqahmed, G. Umesh Suresh, and C. Author, “MEDICAL ROBOTICS AND AUTOMATION,” Borchate Sourabh Shivaji et al, 2013, Accessed: Nov. 18, 2022. [Online]. Available: www.ijmerr.com

[67] “Business Process Model and Notation (BPMN), Version 2.0,” 2010, Accessed: Aug. 03, 2022. [Online]. Available: http://www.omg.org/spec/BPMN/20100501

[68] “Flutter.” Accessed: Aug. 15, 2022. [Online]. Available: https://flutter.dev/

[69] IEEE Robotics and Automation Society. Standing Committee for Standards Activities., Institute of Electrical and Electronics Engineers., and IEEE-SA Standards Board., “IEEE standard ontologies for robotics and automation,” p. 45.

[70] S. R. Fiorini et al., “Extensions to the core ontology for robotics and automation,” Robot Comput Integr Manuf, vol. 33, pp. 3–11, 2015, doi: 10.1016/j.rcim.2014.08.004.

[71] F. D. Davis and V. Venkatesh, “Toward Preprototype User Acceptance Testing of New Information Systems: Implications for Software Project Management,” IEEE Trans Eng Manag, vol. 51, no. 1, 2004, doi: 10.1109/TEM.2003.822468.

[72] C. Z. Sprenger and T. Ribeaud, "Robotic Process Automation with Ontology-Enabled Skill-Based Robot Task Model and Notation (RTMN)," in Proc. 2nd Int. Conf. Robotics, Automation and Artificial Intelligence (RAAI), Singapore, Dec. 9–11, 2022, pp. 15–20. [CrossRef].

[73] Y. Li and S. S. Ge, "Human-Robot Collaboration Based on Motion Intention Estimation," IEEE/ASME Trans. Mechatron., vol. 19, pp. 1007–1014, 2014. [CrossRef].

[74] A. Weiss, A. K. Wortmeier, and B. Kubicek, "Cobots in Industry 4.0: A Roadmap for Future Practice Studies on Human-Robot Collaboration," IEEE Trans. Hum. Mach. Syst., vol. 51, pp. 335–345, 2021. [CrossRef].

[75] N. Lubold, E. Walker, and H. Pon-Barry, "Effects of voice-adaptation and social dialogue on perceptions of a robotic learning companion," in Proc. HRI’16: 11th ACM/IEEE Int. Conf. Human Robot Interaction, Christchurch, New Zealand, Mar. 7–10, 2016.

[76] Institute of Electrical and Electronics Engineers, A Special Project of the IEEE Region 3 Strategic Planning Committee, Piscataway, NJ, USA, 2015.

[77] A. Freedy, E. DeVisser, G. Weltman, and N. Coeyman, "Measurement of Trust in Human-Robot Collaboration," in Proc. Int. Symp. Collaborative Technologies and Systems, Orlando, FL, USA, May 25, 2007.

[78] IEEE Staff, Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Piscataway, NJ, USA, 2010.

[79] S. Kumar, C. Savur, and F. Sahin, "Survey of Human-Robot Collaboration in Industrial Settings: Awareness, Intelligence, and Compliance," IEEE Trans. Syst. Man Cybern. Syst., vol. 51, pp. 280–297, 2021. [CrossRef].

[80] S. Kock, T. Vittor, B. Matthias, H. Jerregard, M. Kallman, I. Lundberg, and M. Hedelind, "Robot concept for scalable, flexible assembly automation: A technology study on a harmless dual-armed robot," in Proc. IEEE Int. Symp. Assembly and Manufacturing, Tampere, Finland, May 25–27, 2011.

[81] E. Matheson, R. Minto, E. G. G. Zampieri, M. Faccio, and G. Rosati, "Human–Robot Collaboration in Manufacturing Applications: A Review," Robotics, vol. 8, p. 100, 2019. [CrossRef].

[82] A. Kim, "A Shortage of Skilled Workers Threatens Manufacturing’s Rebound." Available: https://www.ge.com/news/reports/a-shortage-of-skilled-workers-threatens-manufacturings-r. [Accessed: Oct. 23, 2023].

[83] A. Vysocky and P. Novak, "Human—Robot Collaboration in Industry," MM Sci. J., pp. 903–906, 2016. [CrossRef].

[84] "International Federation of Robotics." Available: https://ifr.org/. [Accessed: Jul. 18, 2023].

[85] ISO 12100:2010; Safety of Machinery—General Principles for Design—Risk Assessment and Risk Reduction, International Organization for Standardization, Geneva, Switzerland, 2010.

[86] ISO 13849-1:2023; Safety-Related Parts of Control Systems—Part 1: General Principles for Design, International Organization for Standardization, Geneva, Switzerland, 2023.

[87] ISO 13850:2015; Safety of Machinery—Emergency Stop Function—Principles for Design, International Organization for Standardization, Geneva, Switzerland, 2015.

[88] ISO 13851:2019; Safety of Machinery—Two-Hand Control Devices—Principles for Design and Selection, International Organization for Standardization, Geneva, Switzerland, 2019.

[89] ISO 10218-1:2011; Robots and Robotic Devices—Safety Requirements for Industrial Robots—Part 1: Robots, International Organization for Standardization, Geneva, Switzerland, 2011.

[90] ISO 10218-2:2011; Robots and Robotic Devices—Safety Requirements for Industrial Robots—Part 2: Robot Systems and Integration, International Organization for Standardization, Geneva, Switzerland, 2011.

[91] EC 62061; Safety of Machinery Functional Safety of Safety-Related Electrical, Electronic and Programmable Electronic Control System, International Electrotechnical Commission, London, UK, 2005.

[92] ISO/TS 15066:2016; Robots and Robotic Devices—Collaborative Robots, International Organization for Standardization, Geneva, Switzerland, 2016.

[93] R. Müller, M. Vette, and A. Geenen, "Skill-Based Dynamic Task Allocation in Human-Robot-Cooperation with the Example of Welding Application," Procedia Manuf., vol. 11, pp. 13–21, 2017. [CrossRef].

[94] L. Wang, R. Gao, J. Váncza, J. Kruger, X. V. Wang, S. Makris, and G. Chryssolouris, "Symbiotic Human-Robot Collaborative Assembly," CIRP Ann., vol. 68, pp. 701–726, 2019. [CrossRef].

[95] S. Thiemermann, Direkte Mensch-Roboter-Kooperation in Der Kleinteilemontage Mit Einem SCARA-Roboter, Ph.D. dissertation, Univ. Stuttgart, Stuttgart, Germany, 2004.

[96] R. Müller, M. Vette, and O. Mailahn, "Process-Oriented Task Assignment for Assembly Processes with Human-Robot Interaction," Procedia CIRP, vol. 44, pp. 210–215, 2016. [CrossRef].

[97] V. Wang, Z. Kemény, J. Váncza, and L. Wang, "Human-Robot Collaborative Assembly in Cyber-Physical Production: Classification Framework and Implementation," CIRP Ann., vol. 66, pp. 5–8, 2017. [CrossRef].

[98] J. Krü, T. K. Lien, and A. Verl, "Cooperation of Human and Machines in Assembly Lines," CIRP Ann., vol. 58, pp. 628–646, 2009. [CrossRef].

[99] S. A. White, Introduction to BPMN, Ibm Cooperation, New York, NY, USA, 2004.

[100] R. Lindorfer and R. Froschauer, ""Towards user-oriented programming of skill-based Automation Systems using a domain-specific Meta-Modelling Approach,"" in Proc. IEEE 17th Int. Conf. Industrial Informatics (INDIN), Helsinki, Finland, Jul. 22–25, 2019, pp. 655–660. [CrossRef].

[101] M. Pantano, Y. Pavlovskyi, E. Schulenburg, K. Traganos, S. Ahmadi, D. Regulin, D. Lee, J. Saenz, "Novel Approach Using Risk Analysis Component to Continuously Update Collaborative Robotics Applications in the Smart," Connected Factory Model. Appl. Sci. 2022, 12, 5639. [CrossRef]

[102] Mahulea, C.; Grau, A.; Lo Bello, L. The 24th IEEE International Conference on Emerging Technologies and Factory Automation Held in Zaragoza, Spain [Society News]. IEEE Ind. Electron. Mag. 2019, 13, 127–128. [CrossRef]

[103] Schmidbauer, C.; Schlund, S.; Ionescu, T.B.; Hader, B. Adaptive Task Sharing in Human-Robot Interaction in Assembly. In Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Singapore, 14 December 2020; pp. 546–550.

[104] Guiochet, J. Hazard Analysis of Human-Robot Interactions with HAZOP-UML. Saf. Sci. 2016, 84, 225–237. [CrossRef]

[105] Martin-Guillerez, D.; Guiochet, J.; Powell, D.; Zanon, C. A UML-Based Method for Risk Analysis of Human-Robot Interactions. In Proceedings of the 2nd International Workshop on Software Engineering for Resilient Systems, New York, NY, USA, 15 April 2010; pp. 32–41.

[106] Guiochet, J.; Motet, G.; Baron, C.; Boy, G. Toward a Human-Centered UML for Risk Analysis Application to a Medical Robot. In Human Error, Safety and Systems Development; Springer: Berlin/Heidelberg, Germany, 2004; pp. 177–191.

[107] Guiochet, J.; Hoang, Q.A.D.; Kaaniche, M.; Powell, D. Model-Based Safety Analysis of Human-Robot Interactions: The MIRAS Walking Assistance Robot. In Proceedings of the IEEE International Conference on Rehabilitation Robotics (ICORR), Seattle, WA, USA, 24–26 June 2013.

[108] Von Borstel, F.D.; Villa-Medina, J.F.; Gutiérrez, J. Development of Mobile Robots Based on Wireless Robotic Components Using UML and Hierarchical Colored Petri Nets. J. Intell. Robot. Syst. Theory Appl. 2022, 104, 70. [CrossRef]

[109] Carroll, L.; Tondu, B.; Baron, C.; Geffroy, J.C. UML Framework for the Design of Real-Time Robot Controllers; Springer: Berlin/Heidelberg, Germany, 1999.

[110] Verband der Elektrotechnik, E.; Institute of Electrical and Electronics Engineers. Modelling Robot Assembly Tasks in Manufacturing Using SysML. In Proceedings of the 41st International Symposium on Robotics, Munich, Germany, 2–3 June 2014.

[111] Ohara, K.; Takubo, T.; Mae, Y.; Arai, T. SysML-Based Robot System Design for Manipulation Tasks. In Proceedings of the 5th International Conference on the Advanced Mechatronics, Shenzhen, China, 18–21 December 2020.

[112] Candell, R.; Kashef, M.; Liu, Y.; Foufou, S. A SysML Representation of the Wireless Factory Work Cell: Enabling Real-Time Observation and Control by Modelling Significant Architecture, Components, and Information Flows. Int. J. Adv. Manuf. Technol. 2019, 104, 119–140. [CrossRef] [PubMed]

[113] Avram, O.; Baraldo, S.; Valente, A. Generalized Behavior Framework for Mobile Robots Teaming with Humans in Harsh Environments. Front. Robot. AI 2022, 9, 898366. [CrossRef] [PubMed]

[114] Peterson, J.L. Petri Nets*. Comput. Surv. 1977, 9, 223–252. [CrossRef]

[115] Casalino, A.; Zanchettin, A.M.; Piroddi, L.; Rocco, P. Optimal Scheduling of Human-Robot Collaborative Assembly Operations with Time Petri Nets. IEEE Trans. Autom. Sci. Eng. 2019, 18, 70–84. [CrossRef]

[116] Chao, C.; Thomaz, A. Timed Petri Nets for Fluent Turn-Taking over Multimodal Interaction Resources in Human-Robot Collaboration. Int. J. Rob. Res. 2016, 35, 1330–1353. [CrossRef]

[117] Chao, C.; Thomaz, A. Timing in Multimodal Turn-Taking Interactions: Control and Analysis Using Timed Petri Nets. J. Hum. Robot. Interact. 2012, 1, 4–25. [CrossRef]

[118] Institute of Electrical and Electronics Engineers. Proceedings of the ICRA 2014—IEEE International Conference on Robotics and Automation; IEEE: Piscataway, NJ, USA, 2014.

[119] R. E. Yagoda and M. D. Coovert, How to Work and Play with Robots: An Approach to Modelling Human-Robot Interaction. Comput. Hum. Behav. 2012, 28, 60–68. [CrossRef]

[120] Casalino, A.; Cividini, F.; Zanchettin, A.M.; Piroddi, L.; Rocco, P. Human-Robot Collaborative Assembly: A Use-Case Application; Elsevier B.V.: Amsterdam, The Netherlands, 2018; Volume 51, pp. 194–199.

[121] Völzer, H. An Overview of BPMN 2.0 and Its Potential Use. In Business Process Modelling Notation; Lecture Notes in Business Information Processing; Springer: Berlin/Heidelberg, Germany, 2010; Volume 67, pp. 14–15. [CrossRef]

[122] ISO—International Organization for Standardization. Available online: https://www.iso.org/obp/ui/en/#iso:std:iso:10218:-1:ed-2:v1:en (accessed on 7 November 2023).

[123] Caiazzo, C.; Nesti´c, S.; Savkovi´c, M. A Systematic Classification of Key Performance Indicators in Human-Robot Collaboration. Lect. Notes Netw. Syst. 2023, 562, 479–489.

[124] Flutter. Available online: https://flutter.dev/ (accessed on 15 August 2022).

[125] Dartros|Dart Package. Available online: https://pub.dev/packages/dartros (accessed on 15 August 2022).

[126] T. Ribeaud and C. Z. Sprenger, Behavior Trees Based Flexible Task Planner Built on ROS2 Framework. In Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, Stuttgart, Germany, 6–9 September 2022. [CrossRef]

[127] Pang, W.; Gu, W.; Li, H. Ontology-Based Task Planning for Autonomous Unmanned System: Framework and Principle; Journal of Physics: Conference Series; IOP Publishing Ltd.: Bristol, UK, 2022; Volume 2253.

[128] Davis, F.D.; Venkatesh, V. Toward Preprototype User Acceptance Testing of New Information Systems: Implications for Software Project Management. IEEE Trans. Eng. Manag. 2004, 51, 31–46. [CrossRef]

[129] Davis, F.D. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. Manag. Inf. Syst. Q. 1989, 13, 319–339. [CrossRef]

[130] Hancock, P.A.; Billings, D.R.; Schaefer, K.E. Can You Trust Your Robot? Ergon. Des. Q. Hum. Factors Appl. 2011, 19, 24–29. [CrossRef]

[131] Norman, D. The Design of Everyday Things; Basic Books: New York, NY, USA, 2013.

[132] Wickens, C.D.; Gordon, S.E.; Liu, Y.; Lee, J. An Introduction to Human Factors Engineering; Pearson Prentice Hall: Upper Saddle River, NJ, USA, 2004; Volume 2.

[133] Asaro, P.M. What Should I Want from a Robot Ethic? In Machine Ethics and Robot Ethics; Routledge: London, UK, 2020; pp. 87–94.

[134] M. Pantano, T. Eiband, and D. Lee, “Modularization of Skill Ontologies for Industrial Robots,” in 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE), 2022, pp. 2355–2362.

[135] IEEE, “IEEE standard ontologies for robotics and automation,” p. 45, 2015.

[136] I. Niles and A. Pease, “Towards a standard upper ontology,” Formal Ontology in Information Systems: Collected Papers from the Second International Conference, pp. 2–9, 2001, doi: 10.1145/505168.505170.

[137] M. R. Pedersen et al., “Robot skills for manufacturing: From concept to industrial deployment,” Robot Comput Integr Manuf, vol. 37, pp. 282–291, Feb. 2016, doi: 10.1016/J.RCIM.2015.04.002.

[138] A. Nilsson, R. Muradore, K. Nilsson, and P. Fiorini, “Ontology for robotics: A roadmap,” 2009.

[139] Guilhem Buisan, Guillaume Sarthou, Arthur Bit-Monnot, Aur´elie Clodic, and Rachid Alami, Efficient, Situated and Ontology based Referring Expression Generationfor Human-Robot collaboration. IEEE, 2020.

[140] A. Umbrico, A. Cesta, and A. Orlandini, “Deploying Ontology-based Reasoning in Collaborative Manufacturing,” 2022, Accessed: Sep. 05, 2022. [Online]. Available: https://sharework-project.eu

[141] G. Sarthou, A. Clodic, and R. Alami, “Ontologenius: A long-term semantic memory for robotic agents; Ontologenius: A long-term semantic memory for robotic agents,” 2019. doi: 10.0/Linux-x86_64.

[142] J. I. Olszewska et al., “Ontology for autonomous robotics,” RO-MAN 2017 - 26th IEEE International Symposium on Robot and Human Interactive Communication, vol. 2017-January, pp. 189–194, Dec. 2017, doi: 10.1109/ROMAN.2017.8172300.

[143] A. Yuguchi et al., “Toward robot-agnostic home appliance operation: a task execution framework using motion primitives, ontology, and GUI,” Advanced Robotics, vol. 36, no. 11, pp. 548–565, 2022, doi: 10.1080/01691864.2022.2070422.

[144] M. Tenorth and M. Beetz, “KNOWROB - Knowledge processing for autonomous personal robots,” 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 4261–4266, Dec. 2009, doi: 10.1109/IROS.2009.5354602.

[145] M. Weser, J. Bock, S. Schmitt, A. Perzylo, and K. Evers, An Ontology-Based Metamodel for Capability Descriptions. 2020.

[146] A. Perzylo et al., “Capability-based semantic interoperability of manufacturing resources: A BaSys 4.0 perspective,” in IFAC-PapersOnLine, Elsevier B.V., Sep. 2019, pp. 1590–1596. doi: 10.1016/j.ifacol.2019.11.427.

[147] P. M. Schäfer et al., “Flexible Robotic Assembly Based on Ontological Representation of Tasks, Skills, and Resources,” 2021. [Online]. Available: https://github.com/HajoRijgersberg/OM/

[148] A. Saxena, A. Jain, O. Sener, A. Jami, D. K. Misra, and H. S. Koppula, “RoboBrain: Large-Scale Knowledge Engine for Robots,” Dec. 2014, [Online]. Available: http://arxiv.org/abs/1412.0691

[149] R. de A. Falbo, “SABiO: Systematic Approach for Building Ontologies,” ONTO.COM/ODISE@FOIS, 2014.

[150] M. F. Caro, M. T. Cox, and R. E. Toscano-Miranda, “A Validated Ontology for Metareasoning in Intelligent Systems,” J Intell, vol. 10, no. 4, Dec. 2022, doi: 10.3390/JINTELLIGENCE10040113.

[151] C. Zhang Sprenger and T. Ribeaud, “Robotic Process Automation with Ontology-enabled Skill-based Robot Task Model and Notation (RTMN),” in IEEE International Conference on Robotics, Automation and Artificial Intelligence (RAAI 2022), Singapore, 2022. doi: 10.24451/arbor.18794.

[152] C. Zhang Sprenger, J. A. Corrales Ramón, and N. Urs Baier, “RTMN 2.0—An Extension of Robot Task Modeling and Notation (RTMN) Focused on Human–Robot Collaboration,” Applied Sciences 2024, Vol. 14, Page 283, vol. 14, no. 1, p. 283, Dec. 2023, doi: 10.3390/APP14010283.

[153] M. Ghallab et al., “PDDL-The Planning Domain Definition Language Domain-specific Insight Graphs (DIG) View project Ability-Based Design View project SEE PROFILE PDDL | The Planning Domain Deenition Language,” 1998. Accessed: Mar. 01, 2023. [Online]. Available: https://www.researchgate.net/publication/2278933

[154] S. Borgo, A. Cesta, A. Orlandini, and A. Umbrico, “Knowledge-based adaptive agents for manufacturing domains,” Eng Comput, vol. 35, no. 3, pp. 755–779, Jul. 2019, doi: 10.1007/s00366-018-0630-6.

[155] C. Schou, R. Skovgaard Andersen, D. Chrysostomou, S. Bøgh, and O. Madsen, “Skill-based instruction of collaborative robots in industrial settings,” 2018, doi: 10.1016/j.rcim.2018.03.008.

[156] E. A. Topp, M. Stenmark, A. Ganslandt, A. Svensson, M. Haage, and J. Malec, "Ontology-Based Knowledge Representation for Increased Skill Reusability in Industrial Robots," in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Oct. 2018. Accessed: Sep. 28, 2022. [Online]. Available: http://kif.cs.lth.se/ontologies/XXX.owl,

[157] "Protégé," Stanford Center. Accessed: May 09, 2023. [Online]. Available: https://protege.stanford.edu/

[158] D. Paulius and Y. Sun, "A Survey of Knowledge Representation in Service Robotics," Rob Auton Syst, vol. 118, pp. 13–30, Aug. 2019, doi: 10.1016/j.robot.2019.03.005.

[159] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in Workshops at the IEEE International Conference on Robotics and Automation (ICRA), 2009.

[160] "ROS: Home", ros.org. https://www.ros.org/ (accessed: June 4, 2022).

[161] M. Cashmore, M. Fox, D. Long, D. Magazzeni, B. Ridder, A. Carrera, N. Palomeras, N. Hurtós, M. Carreras, "ROSPlan: Planning in the Robot Operating System," in Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling.

[162] F. Martín Rico, M. Morelli, H. Espinoza, F. J. Rodríguez-Lera, V. Matellán Olivera, "Optimized execution of PDDL plans using behavior trees," in Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, (2021), 1584-1586.

[163] M. Colledanchise, P. Ögren, "Behavior Trees in Robotics and AI: An Introduction," in Chapman & Hall/CRC Artificial Intelligence and Robotics Series 2018.

[164] S. Macenski, F. Martin, R. White, J. G. Clavero, "The Marathon 2: A Navigation System," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

[165] D. Faconti, "MOOD2Be: Models and Tools to design Robotic Behaviors," in 2017

[166] Groot: ROBMOSYS Wiki", robmosys.eu. https://robmosys.eu/wiki/baseline:environment_tools:groot (accessed: June 4, 2022)

[167] "ZeroMQ", zeromq.org. https://zeromq.org/ (accessed: June 4, 2022)

[168] "GitHub - ros2/ros1_bridge: ROS 2 package that provides bidirectional communication between ROS 1 and ROS 2", github.com https://github.com/ros2/ros1_bridge (accessed: June 4, 2022)

[169] "Actionlib - ROS Wiki", ros.org. http://wiki.ros.org/actionlib (accessed: June 4, 2022)

[170] N. Urs Baier, J. Zovi, "Flexible Robot Programming using Solid Edge's "Alternative Assemblies"," in Athens Journal of Technology & Engineering, (2021), 1-16

[171] L. Ou, X. Xu, "Relationship matrix based automatic assembly sequence generation from a CAD model," in CAD Computer Aided Design 45 (2013), 1053-1067

[172] B. Deepak, G. Bala Murali, M. Bahubalendruni, B. Biswal, "Assembly sequence planning using soft computing methods: A review," in Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering, (2019), 653-683, 233(3)

[173] Protégé, a free, open-source ontology editor and framework for building intelligent systems. https://protege.stanford.edu/ (accessed: July 22, 2022).

[174] P. Eyerich, R. Mattmüller, G. Röger, "Using the Context-enhanced Additive Heuristic for Temporal and Numeric Planning," in Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS 2009), 2009

[175] A. Coles, A. Coles,M. Fox,D. Long, "Forward-Chaining Partial-Order Planning," in Proceedings of the Twentieth International Conference on Automated Planning and Scheduling (ICAPS 2010)

[176] What is DDS? https://www.dds-foundation.org/what-is-dds3/ (accessed: July 22, 2022)

12 ACKNOWLEDGMENT

I would like to express my deep gratitude to my supervisors Norman Urs Baier and Juan Antonio Corrales Ramón for their patient guidance, encouragement, and critiques of the research work. Special thanks to my work colleagues for their technical support and their great collaboration. I would also like to thank the EU project ACROBA for providing the opportunity and funding for this research. I would like to extend my thanks to the ACROBA consortium for their contributions and support with the survey and interviews.

13 INDEX OF FIGURES

Figura 1. Conclusións da revisión da literatura 8
Figura 2. Fase de investigación 9
Figura 3. Deseño de investigación secuencial explicativa 11
Figure 7. Literature Review Finding 17
Figure 8. Research Gap 18
Figure 9. Basic Mixed Methods Research Design [45] 25
Figure 10. Explanatory Sequential Research Design 26
Figure 11. Literature Review Overview 27
Figure 12. Google Form Questionnaires 32
Figure 13. Question 11: What are the problems you have had working with robots? 32
Figure 14. Question 4: What do you expect for a user interface of ACROBA? 33
Figure 15. Question 5 Which of the following task representations do you like the most? 33
Figure 16. Question 6: What do you expect from a user interface for planning robot tasks? 33
Figure 17. Robotic Task Model and Notation (RTMN) 35
Figure 18. RTMN User Interface 37
Figure 19. RTMN Framework 38
Figure 20. Question and Answers for Prototype Validation 40
Figure 21. Answer Bar Chart (see Appendix 15.8 for the clearer version) 41
Figure 22. RTMN modelling elements—Part A: Basic Notations 51
Figure 23. RTMN modelling elements—Part B: HRC tasks 53
Figure 24. RTMN modelling elements—Part C: other notations 54
Figure 25. RTMN 2.0 sequence flow connection rules 55
Figure 26. Combining HRC task types and HRC modes 56
Figure 27. Coexistence Fence Notation 57
Figure 28. Sequential Cooperation SMS Notation 57
Figure 29. Teaching HG Notation 58
Figure 30. Parallel Cooperation SSM Notation 58
Figure 31. Collaboration PFL Notation 59
Figure 32. Minimum separation distance 59
Figure 33. Workspace notation 60
Figure 34. Example of using workspace notation 60
Figure 35. Condition notation 60
Figure 36. Decision notation 61
Figure 37. Example of using decision notation 62
Figure 38. Requirement notation. 63
Figure 39. KPI notation 63
Figure 40.Robotic process notation 64
Figure 41. Process–requirement–KPI example 65
Figure 42. Skill notation. 65
Figure 43. Primitive notation 65
Figure 44. HRC process example—PCB assembly 67
Figure 45. Combined Ontology Development Methodology (Created based on [149] and [150]). 73
Figure 46. Ontology Purpose Identification and Requirements Elicitation (Source: [149]). 74
Figure 47. Ontology Conceptualization and Formalization (Source: [26]) 76
Figure 48. Analysis of Ontology Terms Related to Robotic Task Planning. 77
Figure 49. Skills Analysis on Five Industrial Pilot Lines 79
Figure 50. Ontology Architecture. 81
Figure 51. Taxonomy of the main Concepts of ORPP and Their Relation to the Other Ontologies. 82
Figure 52. ORPP Core Relation Graph 89
Figure 53. Uses of ORPP ontologies 91
Figure 54. Projection of an ontology instantiation—a PCB assembly process 92
Figure 55. Individual HRCTask and its use in ORPP. 93
Figure 56. SPARQL query for PCBAssembly process instantiation. 93
Figure 57. Ontology Query Support Use of Ontology 94
Figure 58. The Demonstrator 95

Figure 59. Initial State 96
Figure 60. Goal State 96
Figure 61. PDDL Ontology Relation Graph 98
Figure 62. ACROBA Architecture Overview 104
Figure 63. Task planner interfaces to other ACROBA modules 106
Figure 64. The Skills Concept 107
Figure 65. Task Planner Link to GUI, Ontology and ROS 108
Figure 66. Experimental Validation with 3 Demonstrations 110
Figure 67. Assembly Process Flow 111
Figure 68. The Graphical User Interface 111
Figure 69. Place Circular-Container in Grid Property Screen 112
Figure 70. Scan Scene Property Screen 113
Figure 71. Plan Movements Property Screen 114
Figure 72. Locate Cylinder in Bin Property Screen 115
Figure 73. Plack Cylinder Property Screen 116
Figure 74. Place Cylinder Property Screen 117
Figure 75. Step1a 118
Figure 76. Step1b 118
Figure 77. Step2 118
Figure 78. Step5a Move to Bin 119
Figure 79. Step5b Locate cylinder 119
Figure 80. Step6 120
Figure 81. Step7 120
Figure 82. Pick 121
Figure 83. Place 121
Figure 84. Task Finish 121
Figure 85. Scenario 2 Initial State 122
Figure 86. Scenario 2 Goal State 123
Figure 87. HRC Task 127
Figure 88. Components of the HRC Task 127
Figure 89. HRC Green Zone 128
Figure 90. HRC Yellow Zone 128
Figure 91. HRC Red Zone 129

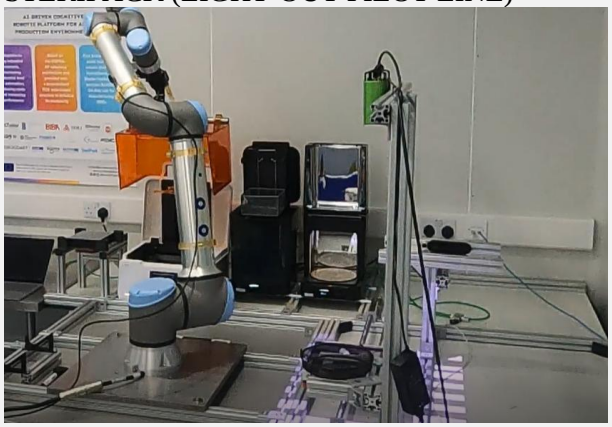
14 INDEX OF TABLES

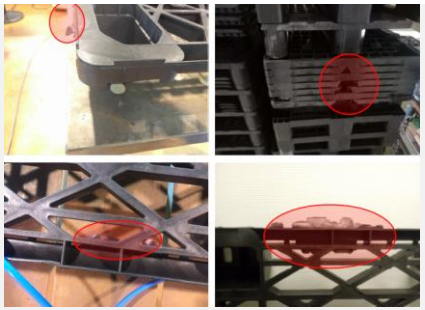
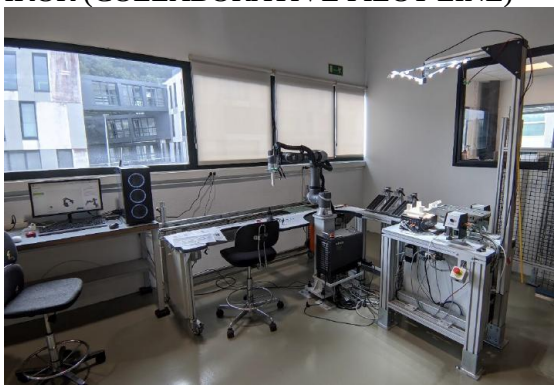
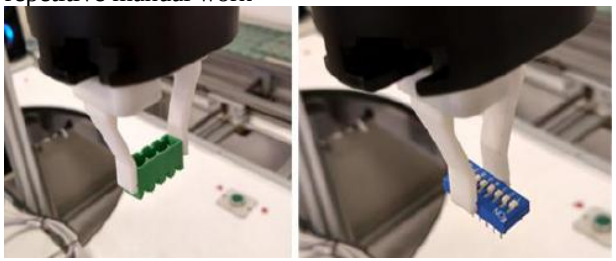
Táboa 1. Detalles da publicación 1 14
Táboa 2. Detalles da publicación 2 14
Táboa 3. Detalles da publicación 3 14
Táboa 4. Detalles da publicación 4 14
Table 5. Interview Questions 39
Table 6. Safety Standards 44
Table 7. Rule Table 61
Table 8. Requirements and KPIs 63
Table 9. Competency Questions and Answers 74
Table 10. Class and Properties 80

15 APPENDICES

15.1 ACROBA USE CASES

The ACROBA use cases are STER, MOSES, CABKA, IKOR and ICPE. They participate in the project with the role of end-user and demonstrator. They implement ACROBA solutions into their production lines and test the performance and flexibility of ACROBA. Industry feedback is given by the use cases throughout the duration of the project. Appendix 15.1 describes the use cases, their challenges and goals.

USE CASE AND ROBOTIC CELL	DESCRIPTION	GOAL AND PRODUCT	CHALLENGE
STERIPACK (LIGHT-OUT PILOT LINE) 	SteriPack Ireland focuses on contract engineering and manufacturing services in the medical device and healthcare industries. In ACROBA it demonstrates the medical device manufacturing pilot line.	To integrate advanced robotic solutions into the production line to support, speed up and enhance the quality of the 3D printing operations. 	• allow non-programmers to model and control robots • automate the entire 3D printing process of medical devices with robots in a light out environment with no human presence • increase product quality • reduce cycle time
MOSES PRODUCTOS (LIGHT-OUT PILOT LINE) 	MOSES PRODUCTOS is a spin-off of AITIIP Technological Centre. It is an expert company in the transformation of Fossil based, Biobased, and Biodegradable materials based on customer need. In ACROBA it focuses on big plastic manufacturing.	To develop a robust quick programming robot system with the capacity to generate fully customized cutting paths for a mass customization concept in urban furniture. 	• Mass customization • reduce robot programming time • increase product quality • reduce cycle time
CABKA(LIGHT-OUT PILOT LINE)	Cabka is a manufacturer of technical plastic products. The company was among	To achieve an efficient deburring of the pallets by locating the defect and only debur the zones affected by the bur. The products with bur are presented in the picture below	• Defect handling (deburring) of multiple pallet types

	the first to produce lightweight plastic pallets for worldwide product transportation. In ACROBA it focuses on plastic pallets defect handling.		• reduce robot programming time • increase product quality • reduce cycle time
IKOR (COLLABORATIVE PILOT LINE) 	IKOR Sistemas Electrónicos is a global company committed to innovation that provides a total service for the design and manufacture of electronic circuits (EMS). In ACROBA it focuses on electronic components assembly.	To reduce manual labor and automate the PTH electronic components assembly process and to reduce tedious and repetitive manual work 	• flexibly adapt to different components • increase product quality • increase productivity • increase the quality of human workers
ICPE (COLLABORATIVE PILOT LINE) 	ICPE is a Romanian medium enterprise covering a wide range of innovative concerns in the field of electric engineering. In ACROBA it focuses on electric motors manufacturing.	To automate the coil winding, magnet bonding, and coiling bonding process in electric motors manufacturing. 	• flexibly adapt to different operations • increase product quality • reduce labor costs • increase the quality of human workers

15.2 User Questionnaire

Questionnaire for ACROBA User Interface

* Indicates required question

1. Email *

About you

2. Role *

3. Expertise Level *

Mark only one oval.

- Beginner
- Intermediate
- Advanced
- Expert
- Other:

4. Relation to the project *

UI in General

1. What are the users of the ACROBA platform?

Tick all that apply.

- Business users
- Engineers

- [ ] Programmers
- [ ] Manager
- [ ] Other: _____

2. Do the users want to interact with the system? *

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Only when error occurs
- ◯ Other: _____

3. On what device do you expect to manage the control of the robot? *

Mark only one oval.

- ◯ Smartphone / Tablet
- ◯ Laptop / PC
- ◯ HMI
- ◯ Other: _____

4. What do you expect for the user interface of ACROBA? *

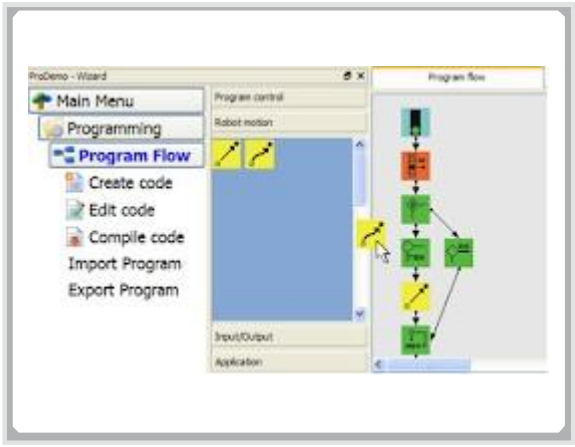
Tick all that apply.

- [ ] One user interface for both planning and execution
- [ ] Separate user interface for planning and execution
- [ ] Graphical user interface
- [ ] Simple texts and buttons
- [ ] Easy to use
- [ ] No user interface needed
- [ ] Other: _____

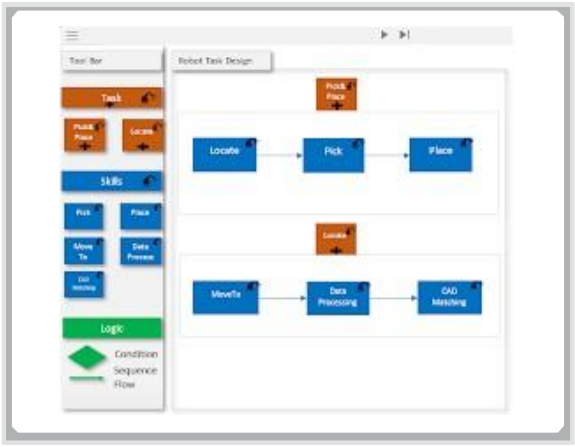
UI for Robot Tasks

5. Which of the following task representations do you like the most? *

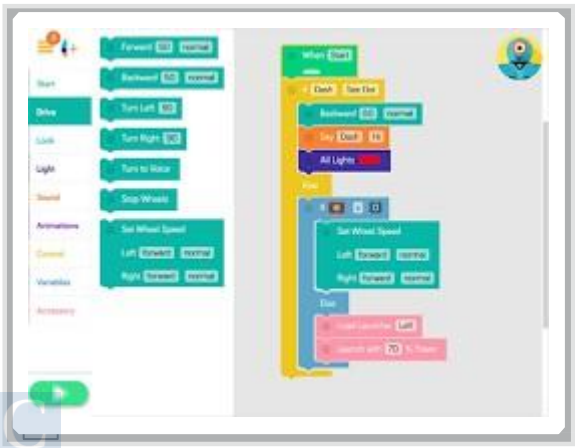
Tick all that apply.



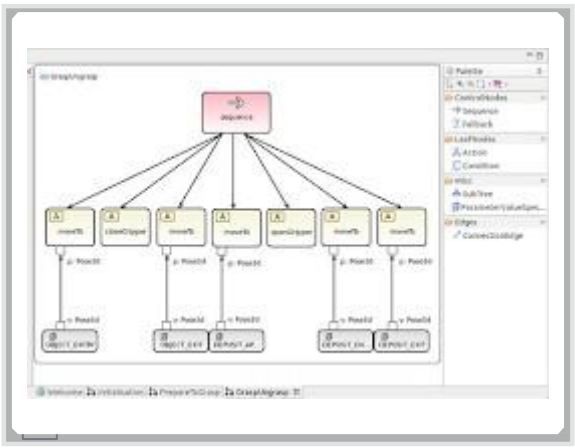
b



g



e



f

6. What do you expect from a user interface for planning robot tasks? *

Tick all that apply.

- [ ] Drag and drop robot tasks to planning
- [ ] Run simulation for different scenarios
- [ ] Present different optimizations to choose from
- [ ] User friendly
- [ ] Graphical representation
- [ ] Other: _____

7. What do you expect from a user interface when executing the robot tasks? *

Tick all that apply.

- [ ] An overview of the task execution
- [ ] Control the robot tasks execution
- [ ] Provide additional information from different systems when error occurs
- [ ] Other: _____

8. What do you want to control the robot tasks execution? *

Tick all that apply.

- [ ] go forward
- [ ] backward
- [ ] stop
- [ ] repeat
- [ ] run optimization
- [ ] Other: _____

9. What information do you want to see when an error occurs? *

Tick all that apply.

- [ ] Information from the Vision system
- [ ] Information from the safety system
- [ ] Information from robot task
- [ ] Other: _____

10. Do you have experience working with robots in the production line? *

Mark only one oval.

- ◯ Yes Skip to question 15
- ◯ No Skip to section 6 (End)

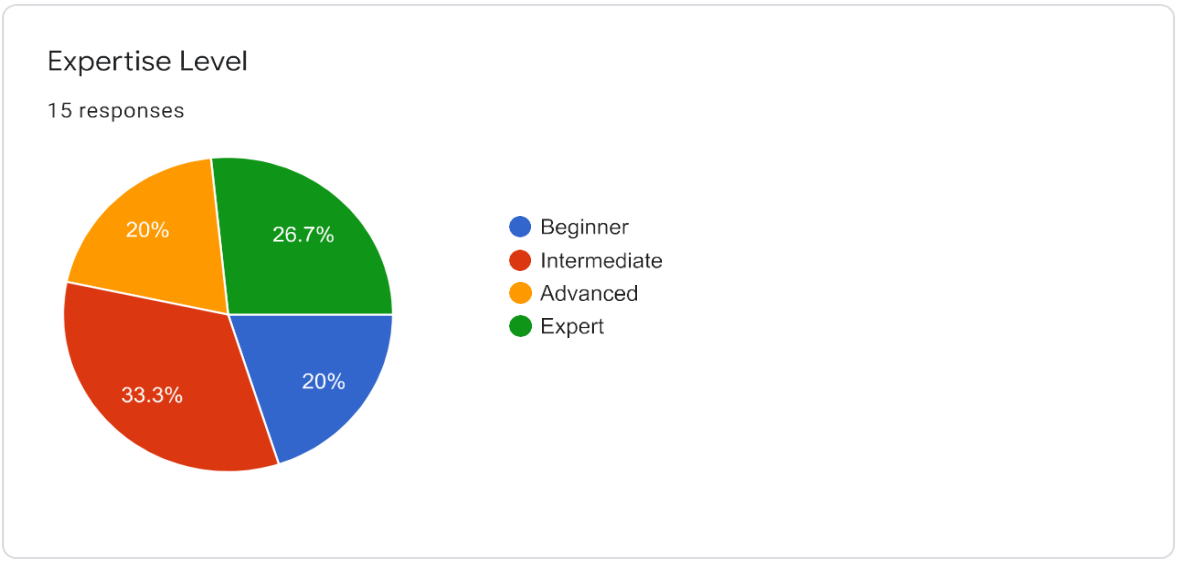
11. What are the problems you have had working with robots? *

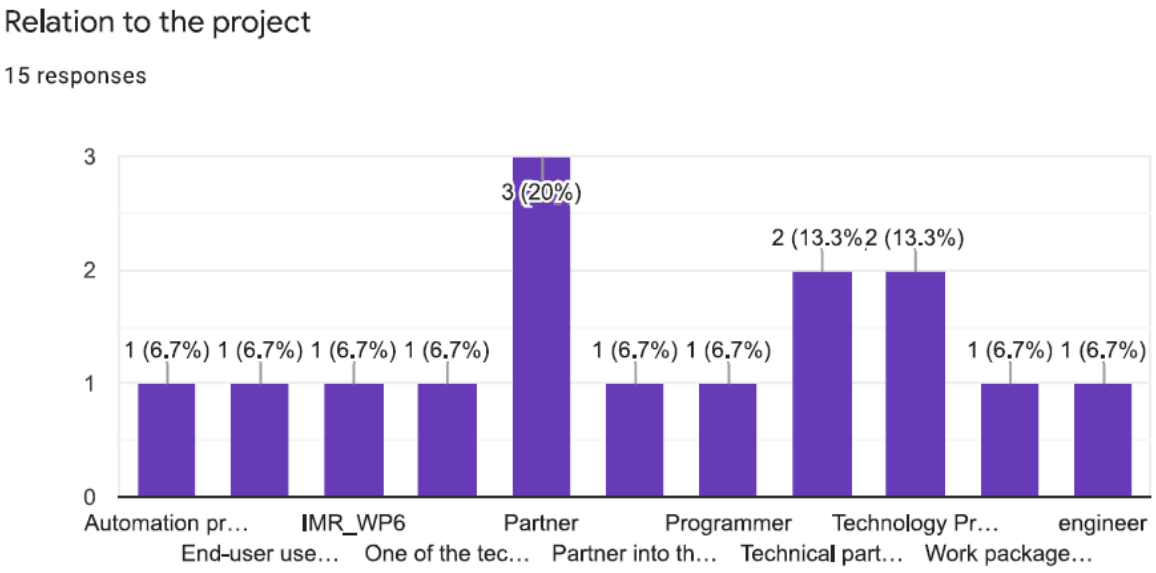
Tick all that apply.

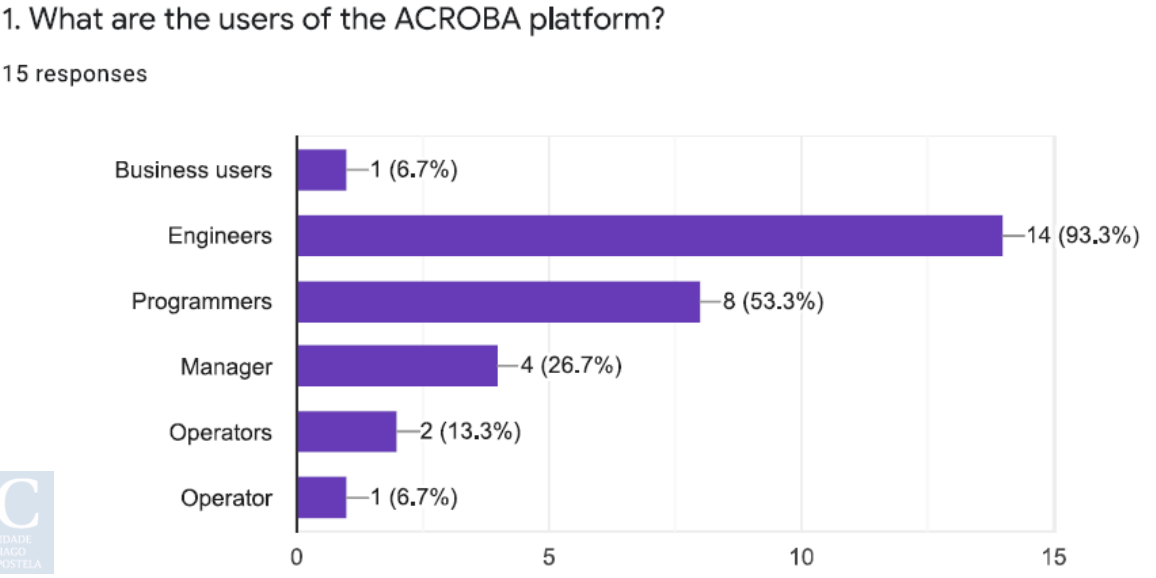
- [ ] When there is an error, it is hard to fix the error
- [ ] Errors / warnings are too technical, not enough information of the system, cannot fix the problem easily
- [ ] Can't control the robot tasks freely
- [ ] Reprogramming robot tasks is very time consuming
- [ ] No overview of the process
- [ ] Lacking information from the vision system or other systems of the tasks performed
- [ ] User interface is too simple
- [ ] User interface is not user friendly
- [ ] Other: _____

15.3 ANSWERS OF THE USER QUESTIONNAIRE

Questionnaire for ACROBA User Interface
15 responses

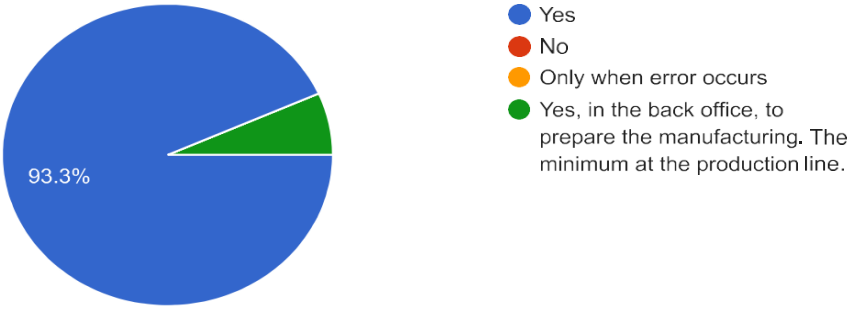






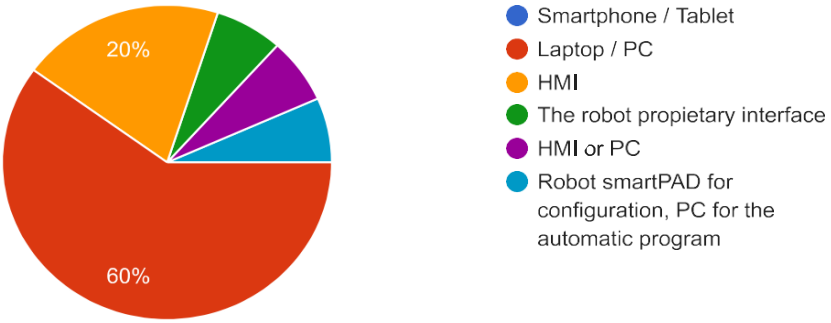
2. Do the users want to interact with the system?

15 responses



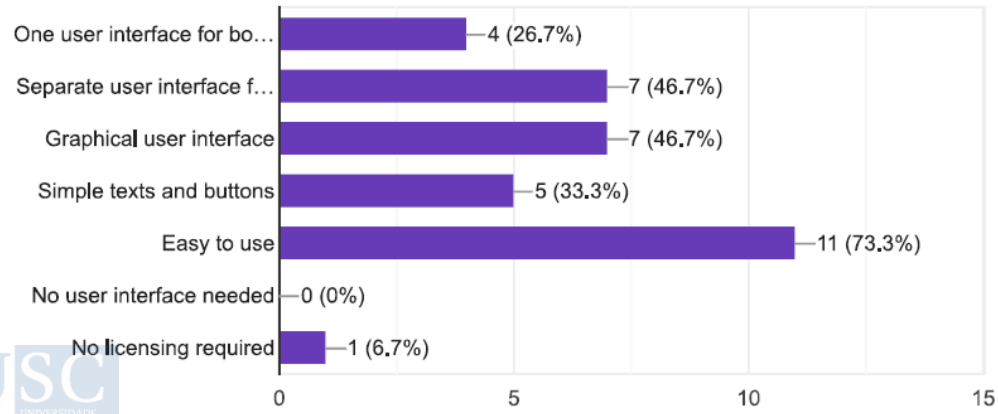
3. On what device do you expect to manage the control of the robot?

15 responses



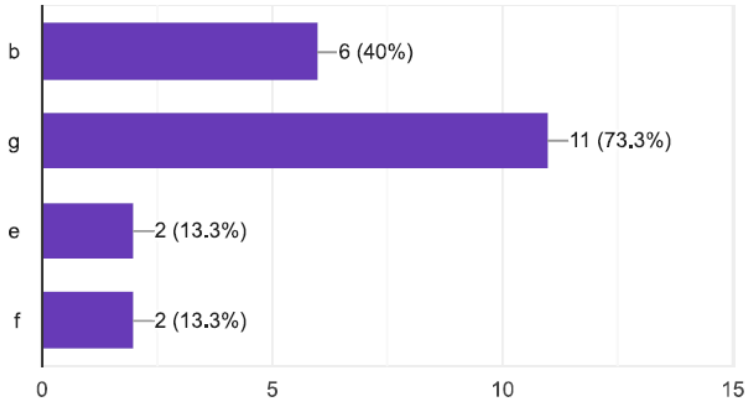
4. What do you expect for a user interface of ACROBA?

15 responses

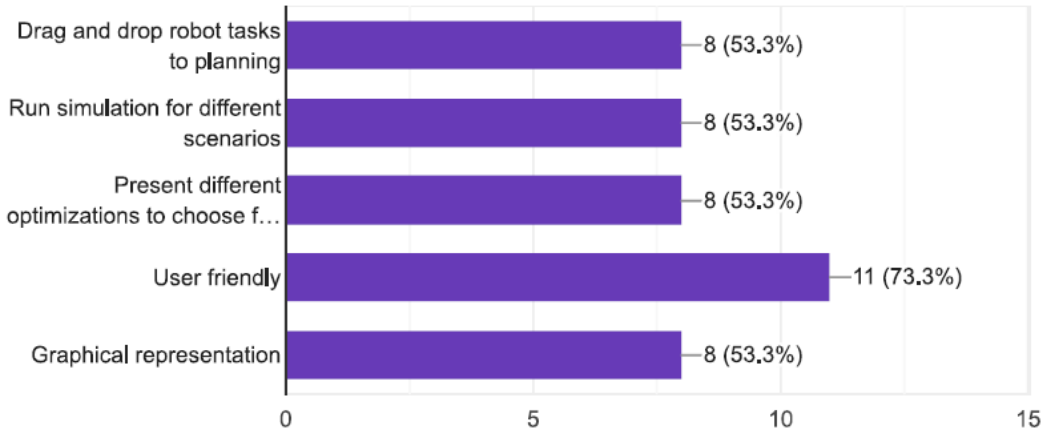


UI for Robot Tasks

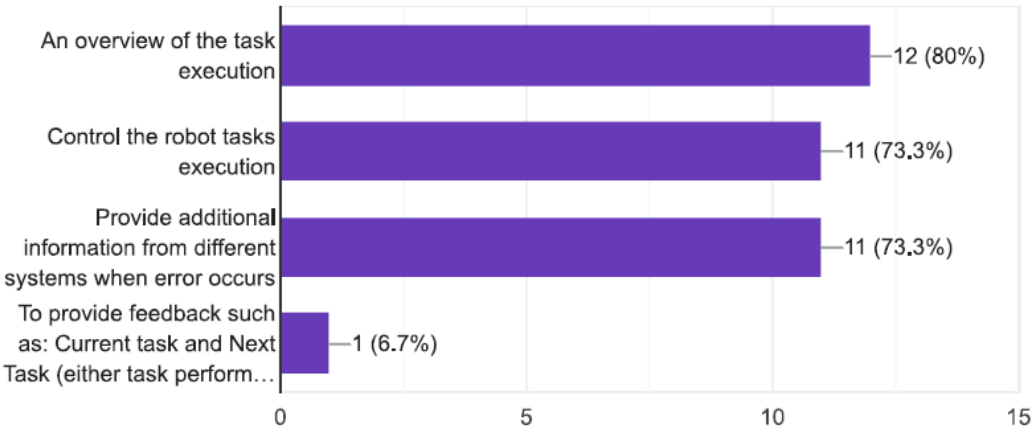
1. Which of the following task representations do you like the most?
15 responses



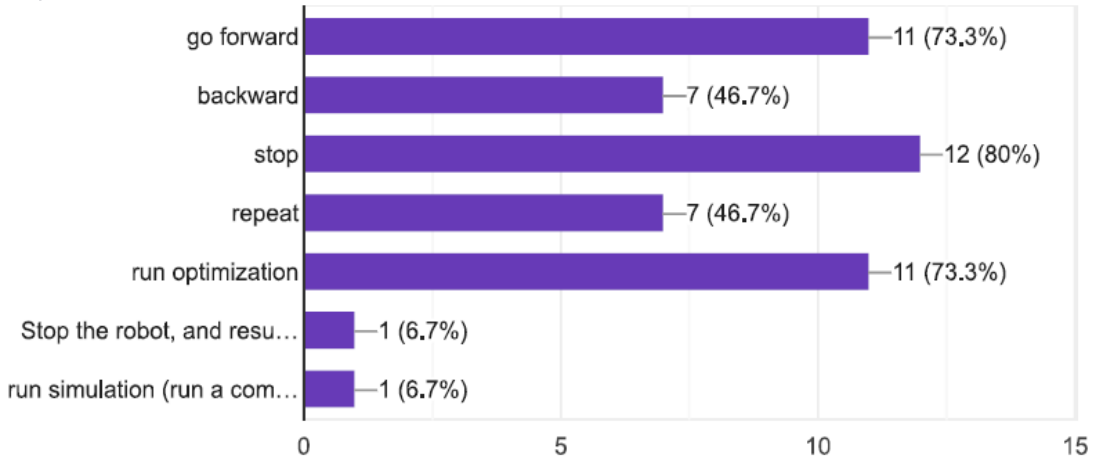
2. What do you expect from a user interface for planning the robot tasks?



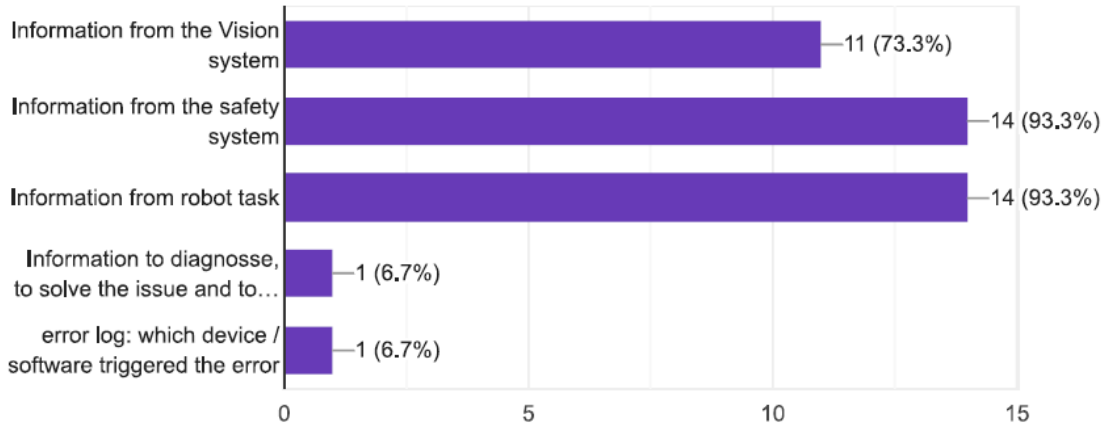
3. What do you expect from a user interface when executing the robot tasks?
15 responses



4. What do you want to control the robot tasks execution?
15 responses

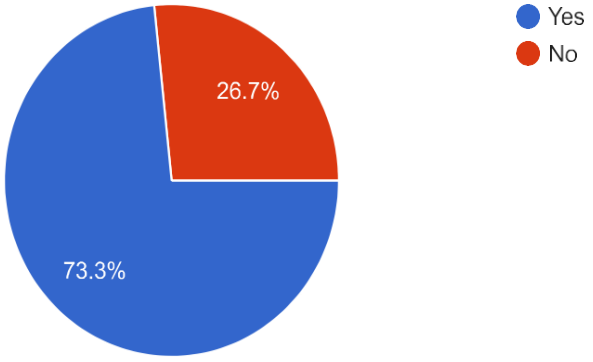


5. What information do you want to see when error occurs?
15 responses

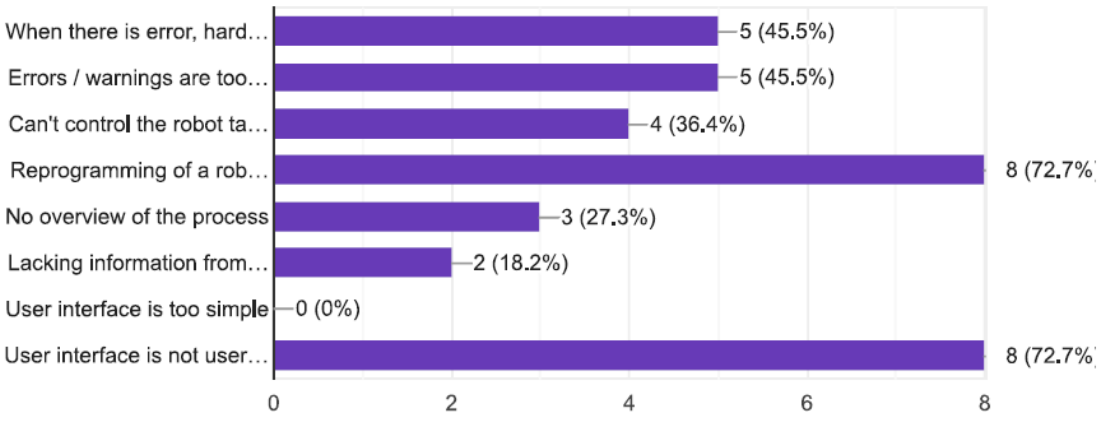


6. Do you have experience working with robots in the production line?

15 responses



7. What are the problems you have had working with robots?
11 responses



15.4 Answers from User Interview

Interview Answers	
IKOR/NUTAI-Technology Partner for IKOR	
Suggestions	Possibility of triggering processes or events by specific conditions. If some variables can be created, it could be useful. Possibility of managing parallel processes For ikor is useful having some flows block such as “follow position on the tray”. Is there a skill for that?
Human role	Operator is on the robotic cell and has only to view the state. Programmers can modify the process, but not creating new skills. Responsible person of the process has the access to the whole functionality of the GUI
Errors	Information is not really detailed, but enough to understand if the process can be restarted, aborted, continued, etc. Log file with information about the error, maybe indicating at which step of the flow the error occurs and at which task/skill.
Validation of the process	Process engineers, A simulation is appreciated.
CABKA	
Suggestions	N/A
Human role	Different levels of access

Errors	If an error occurs, it would be great if the operator can check the type of error and understand where the error is (scanner detects the lack or excess of material) It is better if it is possible to show a drawing instead of a description of the error. Operator probably doesn't read a lot.
Validation of the process	Automation/process engineer Simulation is suitable.
ICPE	
Suggestions	Maybe the color used for human tasks could make difficult reading the text.
Human role	Different levels of access
Errors	For the operator only error message Log file containing more details about the error.
Validation of the process	Process engineers, A simulation is appreciated.
STER	
Suggestions	How to manage data related to the skills (e.g. bin picking) Possibility of managing parallel processes (e.g. detection of human and bin picking) Possibility of visualizing the output of sensors Identify what processes have been already validated.
Human role	Different levels of access Traceability? Insert a login to identify changes and who made them.
Errors	They will send me some aspects. An error code, with a clear message and possibly location/source of error. Safety, process faults and warnings, system status would be the main categories for faults. Depending on error / fault / warning, I could possibly clear it and continue from where I stopped or have a full "reset" (in the case of a safety breach for example) and start from beginning.
Validation of the process	Three teams: R&D design the process, then it goes to the automation engineering team and the to the operation team. In terms of validation, they would like any changes to the operations/tasks to be recorded. Process / skills or any changes would need to be checked by an engineer and validated / approved by a quality engineer.
General Comments	
	It is missing how to use the dummy tool. How would I record a trajectory? How would I load a CAD file for matching? From the PPT I don't know how those boxes are programmed. When you double click on a box, are you able to edit the parameters and conditions? Add some options to customize the interface The system needs to show (preferably graphically) if there is any problem in the process and where the problem is. Differentiate the created and validated skills that can be used from the skills under developments Add features to put the created new skills in the system

15.5 ANSWERS FROM PROTOTYPE VALIDATION

Prototype Validation Comments	
I think I will need technical support to use this system?	This will depend on the level of complexity involved in the configuration of each block
I would like to use this system frequently	I consider it an offline programming tool. I do not expect to use it frequently, but with every article change in production. NEUTRAL. From my point of view, the common usage will be just monitoring the process and checking alarms. But I might need to change the process and create new skills/tasks in uncommon situations.
I think the system is simple and easy to use	For now, it is. It depends on the new features added. I think I will need technical support to use this system: This will depend on the level of complexity involved in the configuration of each block.
I find the system functions are sufficient	Please, consider to add some more blocs. Perhaps Scratch is a good example to follow, particularly regarding the control, variable and operator blocks. https://en.scratch-wiki.info/wiki/Blocks#Stack_blocks
I think the drag and drop elements are well classified	Maybe a search bar?
I think most people can learn this system quickly	I am only seeing a sketch of the solution. It is still premature to answer this question. I believe that the complexity will come when configuring the intrinsic parameters of each block. That's where a good user-friendly interface will be important. For now, yes, it depends on how the rest of the options are implemented.
I find use of this system time-consuming	I think it is a bit early to say about this. Easy processes would be really fast to configure, but if you need to create new skills/tasks it will need a previous study that would consume more time.
I feel confident using this system	Too early, I see it well, but it needs more features and some experience
I think there are a lot of things to learn before I can start using this system	Yes, I would like to know how to manage possible parallel processes, or process triggers. For example, the robot may be running a component sorting process until it is interrupted by the digital signal informing it that there is a PCB on the conveyor. For sure, a minimum learning process is needed. Understand what skills primitives are, how far you can reach, what can be done
I find the colors on the user interface comfortable and well chosen	Having a dark mode is important for a PC application, I like that. Care with bright colors and white-like color font. No other preference.

15.6 VALIDATION FORM FOR GUI VERSION 2.0

GUI Final Validation - User Feedback						
Layout	***Strongly agree***	***Agree***	***Neutral***	***Disagree***	***Strongly disagree***	***Anwer Detail***
do you find the shapes intuitive?	○	○	○	○	○	
do you find the color properiate?	○	○	○	○	○	
do you find the buttons easy to use?	○	○	○	○	○	
do you consider the drag and drop elements easy to find	○	○	○	○	○	
Content						
do you find the categories (process, task, skill) logical?	○	○	○	○	○	
do you find all the functions you need in the GUI?	○	○	○	○	○	
do you find the GUI easy to use?	○	○	○	○	○	
do you find the attributes of the shapes easy to understand?	○	○	○	○	○	
do you think erros are displayed correctly?	○	○	○	○	○	
do you think error message has enough information you need?	○	○	○	○	○	
Control						
do you find the process control buttons useful?	○	○	○	○	○	
do you think the release of the process well defined?	○	○	○	○	○	
do you feel confident to use the GUI?	○	○	○	○	○	
do you think you need training for using the GUI?	○	○	○	○	○	
What do you think can be still improved?						

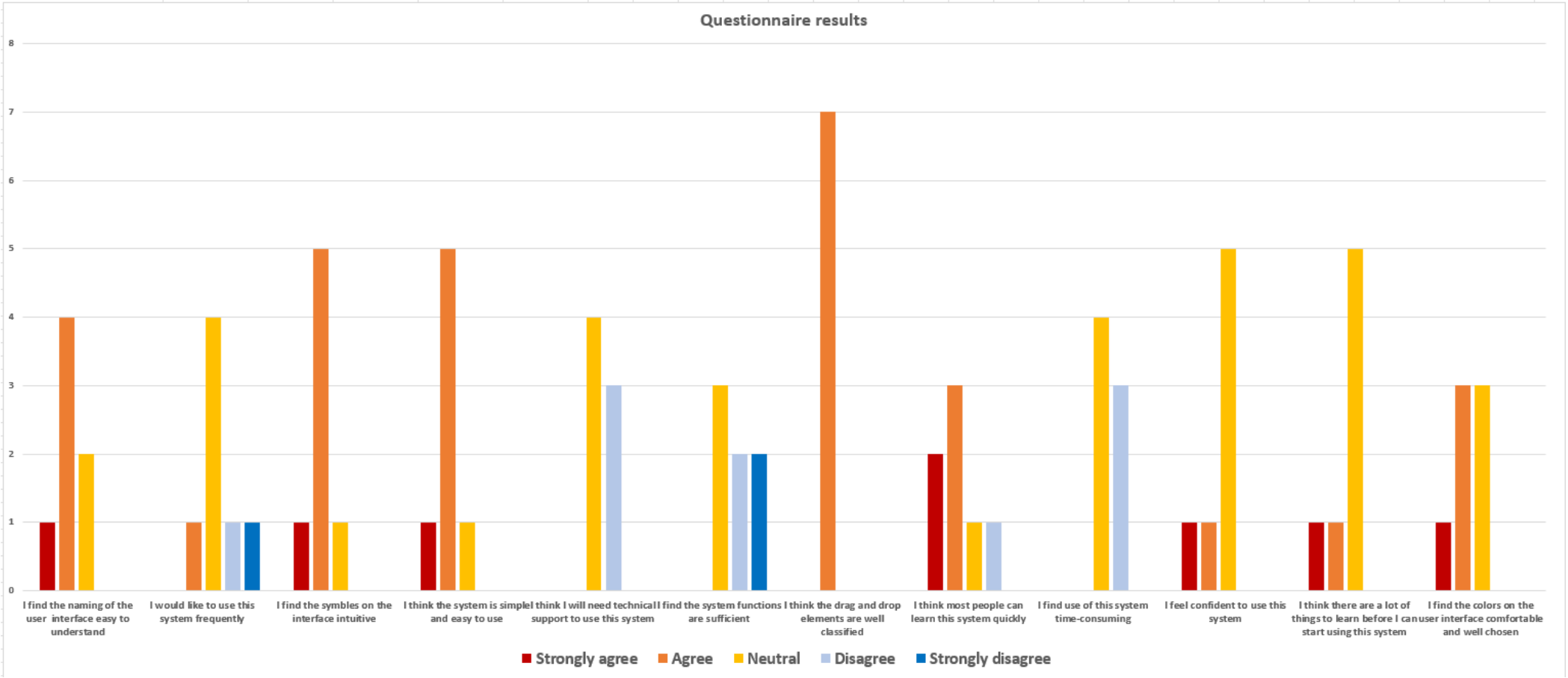
15.7 ONTOLOGY FOR PDDL DOMAIN AND PROBLEM DESCRIPTION

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:owl="http://www.w3.org/2002/07/owl#"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

    <!-- Classes -->
    <owl:Class rdf:about="#Hole"/>
    <owl:Class rdf:about="#Cylinder"/>
    <owl:Class rdf:about="#CircularContainer"/>
    <owl:Class rdf:about="#Bin"/>
    <owl:Class rdf:about="#Colour"/>
    <owl:Class rdf:about="#GoalState"/>

    <!-- Object Properties -->
    <owl:ObjectProperty rdf:about="#hasColour">
        <rdfs:domain rdf:resource="#Cylinder"/>
        <rdfs:range rdf:resource="#Colour"/>
    </owl:ObjectProperty>

    <owl:ObjectProperty rdf:about="#contains">
        <rdfs:domain rdf:resource="#Bin"/>
```

```
    <rdfs:range rdf:resource="#Cylinder"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#empty">
    <rdfs:domain rdf:resource="#Bin"/>
    <!-- Could also be linked to a boolean literal instead -->
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#onGrid">
    <rdfs:domain rdf:resource="#CircularContainer"/>
    <rdfs:range rdf:resource="#Hole"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#placed">
    <rdfs:domain rdf:resource="#Cylinder"/>
    <rdfs:range rdf:resource="#CircularContainer"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="#goalStateReached">
    <rdfs:domain rdf:resource="#GoalState"/>
</owl:ObjectProperty>

<!-- Individuals for Holes -->
<owl:NamedIndividual rdf:about="#h00"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h01"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h02"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h10"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h11"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h12"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h20"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h21"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#h22"><rdf:type rdf:resource="#Hole"/></owl:NamedIndividual>

<!-- Individuals for Cylinders -->
<owl:NamedIndividual rdf:about="#c1"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c2"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c3"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c4"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c5"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c6"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c7"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c8"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c9"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c10"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c11"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c12"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c13"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c14"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c15"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c16"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c17"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#c18"><rdf:type rdf:resource="#Cylinder"/></owl:NamedIndividual>

<!-- Individuals for Circular Containers -->
<owl:NamedIndividual rdf:about="#cc00">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#green"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#cc01">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#yellow"/>
</owl:NamedIndividual>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#cc10">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#yellow"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#cc11">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#green"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#cc12">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#red"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#cc20">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#green"/>
</owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#cc21">
    <rdf:type rdf:resource="#CircularContainer"/>
    <hasColour rdf:resource="#green"/>
</owl:NamedIndividual>

<!-- Individuals for Bins -->
<owl:NamedIndividual rdf:about="#bin-red"><rdf:type rdf:resource="#Bin"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#bin-yellow"><rdf:type rdf:resource="#Bin"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#bin-green"><rdf:type rdf:resource="#Bin"/></owl:NamedIndividual>

<!-- Individuals for Colours -->
<owl:NamedIndividual rdf:about="#red"><rdf:type rdf:resource="#Colour"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#yellow"><rdf:type rdf:resource="#Colour"/></owl:NamedIndividual>
<owl:NamedIndividual rdf:about="#green"><rdf:type rdf:resource="#Colour"/></owl:NamedIndividual>

<!-- Initial State: Bin Contents -->
<rdf:Description rdf:about="#bin-red">
    <contains rdf:resource="#c1"/>
    <contains rdf:resource="#c2"/>
    <contains rdf:resource="#c3"/>
    <contains rdf:resource="#c4"/>
    <contains rdf:resource="#c5"/>
    <contains rdf:resource="#c6"/>
</rdf:Description>

<rdf:Description rdf:about="#bin-yellow">
    <contains rdf:resource="#c7"/>
    <contains rdf:resource="#c8"/>
    <contains rdf:resource="#c9"/>
    <contains rdf:resource="#c10"/>
    <contains rdf:resource="#c11"/>
</rdf:Description>
```

```
    <rdf:Description rdf:about="#bin-green">
        <contains rdf:resource="#c13"/>
        <contains rdf:resource="#c14"/>
        <contains rdf:resource="#c15"/>
        <contains rdf:resource="#c16"/>
        <contains rdf:resource="#c17"/>
    </rdf:Description>

    <!-- Initial State: Cylinder Colours -->
    <rdf:Description rdf:about="#c1"><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#c2"><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#c3"><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#c4"><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#c5"><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#c6"><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#c7"><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#c8"><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#c9"><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#c10"><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#c11"><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#c12"><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#c13"><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#c14"><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#c15"><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#c16"><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#c17"><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#c18"><hasColour rdf:resource="#green"/></rdf:Description>

    <!-- Initial State: Circular Containers on Grid -->
    <rdf:Description rdf:about="#cc00"><onGrid rdf:resource="#h00"/><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#cc01"><onGrid rdf:resource="#h01"/><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#cc10"><onGrid rdf:resource="#h10"/><hasColour rdf:resource="#yellow"/></rdf:Description>
    <rdf:Description rdf:about="#cc11"><onGrid rdf:resource="#h11"/><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#cc12"><onGrid rdf:resource="#h12"/><hasColour rdf:resource="#red"/></rdf:Description>
    <rdf:Description rdf:about="#cc20"><onGrid rdf:resource="#h20"/><hasColour rdf:resource="#green"/></rdf:Description>
    <rdf:Description rdf:about="#cc21"><onGrid rdf:resource="#h21"/><hasColour rdf:resource="#green"/></rdf:Description>

    <!-- Initial State: Cylinders in Circular Containers -->
    <rdf:Description rdf:about="#c13"><placed rdf:resource="#cc10"/></rdf:Description>


    <!-- Initial State: Bin States -->
    <rdf:Description rdf:about="#bin-red"><empty rdf:datatype="xsd:boolean">false</empty></rdf:Description>
    <rdf:Description rdf:about="#bin-yellow"><empty rdf:datatype="xsd:boolean">false</empty></rdf:Description>
    <rdf:Description rdf:about="#bin-green"><empty rdf:datatype="xsd:boolean">false</empty></rdf:Description>



    <!-- Goal State: Example Goal -->
    <rdf:Description rdf:about="#goalState">
        <rdf:type rdf:resource="#GoalState"/>
        <!-- Goal State: Circular Containers on Grid -->
        <rdf:Description rdf:about="#cc00"><onGrid rdf:resource="#h00"/><hasColour rdf:resource="#red"/></rdf:Description>
        <rdf:Description rdf:about="#cc01"><onGrid rdf:resource="#h01"/><hasColour rdf:resource="#yellow"/></rdf:Description>
        <rdf:Description rdf:about="#cc02"><onGrid rdf:resource="#h01"/><hasColour rdf:resource="#green"/></rdf:Description>
        <rdf:Description rdf:about="#cc10"><onGrid rdf:resource="#h10"/><hasColour rdf:resource="#red"/></rdf:Description>
        <rdf:Description rdf:about="#cc11"><onGrid rdf:resource="#h11"/><hasColour rdf:resource="#yellow"/></rdf:Description>
        <rdf:Description rdf:about="#cc12"><onGrid rdf:resource="#h12"/><hasColour rdf:resource="#green"/></rdf:Description>
        <rdf:Description rdf:about="#cc20"><onGrid rdf:resource="#h20"/><hasColour rdf:resource="#red"/></rdf:Description>
        <rdf:Description rdf:about="#cc21"><onGrid rdf:resource="#h21"/><hasColour rdf:resource="#yellow"/></rdf:Description>
        <rdf:Description rdf:about="#cc22"><onGrid rdf:resource="#h21"/><hasColour rdf:resource="#green"/></rdf:Description>

        <!-- Goal State: Cylinders in Circular Containers -->
        <rdf:Description rdf:about="#c1"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c2"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c3"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c7"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c8"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c9"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c13"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c14"><placed rdf:resource="#cc10"/></rdf:Description>
        <rdf:Description rdf:about="#c15"><placed rdf:resource="#cc10"/></rdf:Description>
        <goalStateReached rdf:datatype="xsd:boolean">true</goalStateReached>
    </rdf:Description>

</rdf:RDF>
```

15.8 QUESTIONNAIRE RESULTS


Questionnaire results
8
7
6
5
4
3
2
1
0
I find the naming of the user interface easy to understand
I would like to use this system frequently
I find the symbles on the interface intuitive
I think the system is simple and easy to use
I think I will need technical support to use this system
I find the system functions are sufficient
I think the drag and drop elements are well classified
I think most people can learn this system quickly
I find use of this system time-consuming
I feel confident to use this system
I think there are a lot of things to learn before I can start using this system
I find the colors on the user interface comfortable and well chosen
Strongly agree
Agree
Neutral
Disagree
Strongly disagree

15.9 PUBLICATIONS AND AUTHOR CONTRIBUTIONS

The author has published four articles: 1) Behavior Trees based Flexible Task Planner Built on ROS2 Framework; 2) Robotic Process Automation with Ontology-enabled Skill-based Robot Task Model and Notation (RTMN); 3) RTMN 2.0—An Extension of Robot Task Modelling and Notation (RTMN) Focused on Human–Robot Collaboration; 4) ORPP – An Ontology for Skill-based Robotic Process Planning in Agile Manufacturing. All details are listed below.

15.9.1 Publication 1

Publication Detail	
Title	Behavior Trees based Flexible Task Planner Built on ROS2 Framework
Conference/Journal Name	ETFA Stuttgart September 2022
Author	Congyu Zhang Sprenger, Thomas Ribeaud
Publisher	IEEE
Year of Publication	2022
Status	Published
DOI	https://doi.org/10.1109/ETFA52439.2022.9921683
Reference	T. Ribeaud and C. Zhang Sprenger, "Behavior trees based flexible task planner built on ROS2 framework," in Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation, ETFA, Stuttgart, Germany, Sept. 6–9, 2022. doi: 10.1109/ETFA52439.2022.9921683
Abstract	As modern industrial robotic systems become smarter and more flexible, they are rather tailored for specific, large-scale applications, making their implementation too complex and costly for smaller operators. The ACROBA project aims to develop and demonstrate a novel concept of cognitive robotic platforms based on a modular approach able to be smoothly adapted to virtually any industrial scenario applying agile manufacturing principles. The flexible task planner introduced here is developed as part of this European project. It aims at providing a user-friendly tool allowing the user to reconfigure the robotic cell in quick and cheap way. The main characteristics of the task planner are the behavior tree-based control (which avoids the implementation of complex Functional States Machines), the embedded HMI interface (providing basic functionalities like start, pause, resume, manual selection of skill to run...), the embedded PDDL solver (which allows the task planner to automatically rewrite the task at run time based on factory or sensors inputs), an easy interface with a GUI to design the task manually, and a work-in-progress Assembly Sequence Planner (ASP)

Author Contribution		
Author	Congyu Zhang Sprenger	Thomas Ribeaud
Affiliation	Bern University of Applied Science, School of Engineering and Computer Science, Institute of	Bern University of Applied Science, School of Engineering and Computer Science, Institute of

	I3S, 4 Burgdorf, 3400, Switzerland	I3S, 4 Burgdorf, 3400, Switzerland
Conceptualization (Ideas; formulation or evolution of overarching research goals and aims)	☒	☒
Formal Analysis (Application of statistical, mathematical, computational, or other formal techniques to analyze or synthesize study data)	☒	☒
Data Curation (Management activities to annotate (produce metadata), scrub data and maintain research data (including software code, where it is necessary for interpreting the data itself) for initial use and later reuse)	☐	☐
Methodology (Development or design of methodology; creation of models)	☒	☒
Project Administration (Management and coordination responsibility for the research activity planning and execution)	☒	☒
Supervision (Oversight and leadership responsibility for the research activity planning and execution, including mentorship external to the core team)	☐	☐
Visualization (Preparation, creation and/or presentation of the published work, specifically visualization/data presentation)	☒	☒
Writing – Original Draft Preparation (Creation and/or presentation of the published work, specifically writing the initial draft)	☒	☒
Writing – Review & Editing (Preparation, creation and/or presentation of the published work by those from the original research group, specifically critical review, commentary or revision – including pre- or post-publication stages)	☒	☒

15.9.2 Publication 2

Publication Detail	
Title	Robotic Process Automation with Ontology-enabled Skill-based Robot Task Model and Notation (RTMN)
Conference/Journal Name	IEEE RAAI 2022, Singapore, 09-11 December 2022
Author	Congyu Zhang Sprenger, Thomas Ribeaud
Publisher	IEEE
Year of Publication	2022
Status	Published
DOI	https://doi.org/10.1109/RAAI56146.2022.10092996
Reference	C. Zhang Sprenger and T. Ribeaud, "Robotic process automation with ontology-enabled skill-based robot task model and notation (RTMN)," in IEEE International Conference on Robotics, Automation and Artificial Intelligence (RAAI 2022). doi: 10.24451/arbor.18794.
Abstract	Non-robotic experts are facing challenges in the fast-growing agile production industry. On the one hand robot programming is time-consuming and costly and requires high levels of expertise. On the other hand, current systems are difficult to understand and control. The author proposes bridging this gap by introducing an intuitive way of modelling and programming robotic processes, which enables nonexperts to plan and program robot tasks. A literature review was conducted and then adopted both quantitative and qualitative methods in the project ACROBA to deepen the research on this topic. The author proposes a model-driven framework that combines modelling and programming in a graphical way using RTMN - an ontology-enabled skill-based robot task model and

	notation. Results from the validation process indicate that users find RTMN notations simple to understand and intuitive to use.

Author Contribution		
Author	Congyu Zhang Sprenger	Thomas Ribeaud
Affiliation	Bern University of Applied Science, School of Engineering and Computer Science, Institute of I3S, 4 Burgdorf, 3400, Switzerland	Bern University of Applied Science, School of Engineering and Computer Science, Institute of I3S, 4 Burgdorf, 3400, Switzerland
Conceptualization (Ideas; formulation or evolution of overarching research goals and aims)	☒	☐
Formal Analysis (Application of statistical, mathematical, computational, or other formal techniques to analyze or synthesize study data)	☒	☐
Data Curation (Management activities to annotate (produce metadata), scrub data and maintain research data (including software code, where it is necessary for interpreting the data itself) for initial use and later reuse)	☒	☐
Methodology (Development or design of methodology; creation of models)	☒	☐
Project Administration (Management and coordination responsibility for the research activity planning and execution)	☒	☐
Supervision (Oversight and leadership responsibility for the research activity planning and execution, including mentorship external to the core team)	☐	☐
Visualization (Preparation, creation and/or presentation of the published work, specifically visualization/data presentation)	☒	☐
Writing – Original Draft Preparation (Creation and/or presentation of the published work, specifically writing the initial draft)	☒	☐
Writing – Review & Editing (Preparation, creation and/or presentation of the published work by those from the original research group, specifically critical review, commentary or revision – including pre- or post-publication stages)	☒	☐
Other Contributions: added one figure for this paper		☒

15.9.3 Publication 3

Publication Detail	
Title	RTMN 2.0—An Extension of Robot Task Modelling and Notation (RTMN) Focused on Human–Robot Collaboration
Conference/Journal Name	Applied Sciences Journal Special Issue AI Technologies for Collaborative and Service Robots
Author	Congyu Zhang Sprenger, Juan Antonio Corrales Ramón, Norman Urs Baier

Publisher	MDPI
Year of Publication	2023
Status	Published
DOI	https://doi.org/10.3390/app14010283
Journal Metrics	Open Access, Impact Factor 2.5 (2023), 5-Year Impact Factor: 2.7 (2023), 44/179 (Q1) in Engineering, 125’694 Citations in 2023
Reference	• C. Zhang Sprenger, J. A. Corrales Ramón, and N. U. Baier, "RTMN 2.0—An extension of robot task modeling and notation (RTMN) focused on human–robot collaboration," Applied Sciences, vol. 14, no. 1, p. 283, 2024. doi: 10.3390/APP14010283.
Abstract	This publication describes RTMN 2.0, an extension of the modelling language RTMN. RTMN combines process modelling and robot execution. Intuitive robot programming allows those without programming expertise to plan and control robots through easily understandable predefined modelling notations. These notations achieve no-code programming and serve as templates for users to create their processes via drag-and-drop functions with graphical representations. The design of the graphical user interface is based on a user survey and gaps identified in the literature. The survey was validated through the most influential technology acceptance models, with two major factors: the perceived ease of use and perceived usefulness. While RTMN focuses on the ease of use and flexibility of robot programming by providing an intuitive modelling language, RTMN 2.0 concentrates on Human Robot Collaboration (HRC), which represents the current trend of the industry shift from “mass-production” to “mass-customization”. The biggest contribution that RTMN 2.0 makes is the creation of synergy between HRC modes (based on ISO standards) and HRC task types in the literature. They are modelled as five different HRC task notations: Coexistence Fence, Sequential Cooperation SMS, Teaching HG, Parallel Cooperation SSM, and Collaboration PFL. Both collaboration and safety criteria are defined for each notation. While traditional isolated robot systems in “mass-production” environments provide high payload capabilities and repeatability, they suffer from limited flexibility and dexterity in order to be adapted to the variability of customized products. Therefore, human–robot collaboration is a suitable arrangement to leverage the unique capabilities of both humans and robots for increased efficiency and quality in the new “mass-customization” industrial environments. HRC has made a great impact on the robotic industry: it leads to increased efficiency, reduced costs, and improved productivity, which can be adopted to make up for the skill gap of a shortage of workers in the manufacturing industry. The extension of RTMN 2.0 includes the following notations: HRC tasks, requirements, Key Performance Indicators (KPIs), condition checks and decision making, join/split, and data association. With these additional elements, RTMN 2.0 meets the full range of criteria for agile manufacturing—light-out manufacturing is a manufacturing philosophy that does not rely on human labor.

Author Contribution			
Author	Congyu Zhang Sprenger	Juan Antonio Corrales Ramón	Norman Urs Baier
Affiliation	Bern University of Applied Science, School of Engineering and Computer	Universidade de Santiago de Compo-stela, Centro Singular de Investigación en Tecnoloxías 7 Intelixentes	Bern University of Applied Science, School of Engineering and Computer

	Science, Institute of I3S, 4 Burgdorf, 3400, Switzerland	(CiTIUS), Santiago de Compostela, 15782, Spain	Science, Institute of I3S, 4 Burgdorf, 3400, Switzerland
Conceptualization (Ideas; formulation or evolution of overarching research goals and aims)	☒	☐	☐
Formal Analysis (Application of statistical, mathematical, computational, or other formal techniques to analyze or synthesize study data)	☒	☐	☐
Data Curation (Management activities to annotate (produce metadata), scrub data and maintain research data (including software code, where it is necessary for interpreting the data itself) for initial use and later reuse)	☒	☐	☐
Methodology (Development or design of methodology; creation of models)	☒	☐	☐
Project Administration (Management and coordination responsibility for the research activity planning and execution)	☒	☐	☐
Supervision (Oversight and leadership responsibility for the research activity planning and execution, including mentorship external to the core team)	☐	☒	☒
Visualization (Preparation, creation and/or presentation of the published work, specifically visualization/data presentation)	☒	☐	☐
Writing – Original Draft Preparation (Creation and/or presentation of the published work, specifically writing the initial draft)	☒	☐	☐
Writing – Review & Editing (Preparation, creation and/or presentation of the published work by those from the original research group, specifically critical review, commentary or revision – including pre- or post-publication stages)	☒	☒	☒

15.9.4 Publication 4

Publication Detail	
Title	ORPP – An Ontology for Skill-based Robotic Process Planning in Agile Manufacturing
Conference/Journal Name	Electronics Journal Special Issue Application of Artificial Intelligence in Robotics
Author	Congyu Zhang Sprenger, Juan Antonio Corrales Ramón, Norman Urs Baier
Publisher	MDPI
Year of Publication	2024
Status	Published
DOI	https://doi.org/10.3390/electronics13183666
Journal Metrics	Open Access, Impact Factor 2.6 (2023), 5-Year Impact Factor: 2.6 (2023), 99/158 (Q3) in the "Information Systems, Computer Science" category, 125’694 Citations in 2023
Reference	C. Zhang Sprenger, J. A. Corrales Ramón, and N. U. Baier, "ORPP—An ontology for skill-based robotic process planning in agile manufacturing," Electronics, 2024. doi: 10.3390/electronics13183666.

Abstract	Ontology plays a significant role in AI (Artificial Intelligence) and robotics by providing structured data, reasoning, action understanding, context awareness, knowledge transfer, and semantic learning. The structured framework created by the ontology for knowledge representation is crucial for enabling intelligent behavior in robots. This paper provides a state-of-the-art analysis on the existing ontology approaches and at the same time consolidates the terms in the robotic task planning domain. The major gap identified in the literature is the need to bridge higher-level robotic process management and lower-level robotic control. This gap makes it difficult for operators/non-robotic experts to integrate robots into their production processes as well as evaluate key performance indicators (KPI) of the processes. To fill the gap, the authors propose an ontology for skill-based robotics process planning (ORPP). ORPP not only provides standardization in the robotic process planning in the agile manufacturing domain but also enables non-robotic experts to design and plan their production processes using an intuitive Process-Task-Skill-Primitive structure to control low-level robotic actions. On the performance level, this structure provides traceability of the KPIs down to the robot control level.

Author Contribution			
Author	Congyu Zhang Sprenger	Juan Antonio Corrales Ramón	Norman Urs Baier
Affiliation	Bern University of Applied Science, School of Engineering and Computer Science, Institute of I3S, 4 Burgdorf, 3400, Switzerland	Universidade de Santiago de Compo-stela, Centro Singular de Investigación en Tecnoloxías 7 Intelixentes (CiTIUS), Santiago de Compostela, 15782, Spain	Bern University of Applied Science, School of Engineering and Computer Science, Institute of I3S, 4 Burgdorf, 3400, Switzerland
Conceptualization (Ideas; formulation or evolution of overarching research goals and aims)	☒	☒	☒
Formal Analysis (Application of statistical, mathematical, computational, or other formal techniques to analyze or synthesize study data)	☒	☐	☐
Data Curation (Management activities to annotate (produce metadata), scrub data and maintain research data (including software code, where it is necessary for interpreting the data itself) for initial use and later reuse)	☒	☐	☐
Methodology (Development or design of methodology; creation of models)	☒	☐	☐
Project Administration (Management and coordination responsibility for the research activity planning and execution)	☒	☒	☒
Supervision (Oversight and leadership responsibility for the research activity planning and execution, including mentorship external to the core team)	☐	☒	☒
Visualization (Preparation, creation and/or presentation of the published work, specifically visualization/data presentation)	☒	☐	☐
Writing - Original Draft Preparation (Creation and/or presentation of the published work, specifically writing the initial draft)	☒	☐	☐

Writing – Review & Editing (Preparation, creation and/or presentation of the published work by those from the original research group, specifically critical review, commentary or revision – including pre- or post-publication stages)	☒	☒	☒

This thesis describes the process of my PhD and the research contribution I have made. The main goal of my research is to resolve the challenges that SEMs are facing when implementing robotic solutions in agile manufacturing. The main challenges identified are to gain the rich robotic expertise required to understand the high complexity of existing robotic systems, and to acquire the comprehensive knowledge behind robotic systems, as well as reducing the high programming costs for implementation. My main contributions are to ease these challenges by creating an intuitive modeling language to plan and control robotic processes with the support of ontology. This enables non-robotic experts to flexibly program and reprogram robots as well as acquire relevant knowledge to achieve mass-customization.