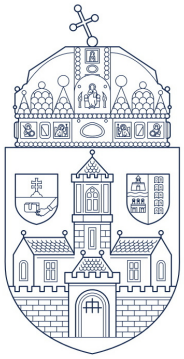
Óbuda University

PhD Thesis



Integration framework for robotics in life science laboratories
Laboratory Automation Plug and Play

Integrációs keretrendszer az élettudományi laboratóriumok robotikájában
Laborautomatizálási Plug and Play

Ádám Wolf

Supervisors: Dr. Péter Galambos, Dr. Károly Széll

Doctoral School of Applied Informatics
and Applied Mathematics

Budapest, 2025

ABSTRACT

Supportive technologies, including sample transportation robots, play a vital role in the expansive field of laboratory automation. Their application spans diverse areas, from academic research within chemistry, materials science, and biotechnology, to sectors in the pharmaceutical industry such as Research and Development (R&D), Quality Control (QC), and process control, reaching into healthcare and food industries. Laboratory automation significantly enhances the efficiency and reproducibility of analytical methods and experimental procedures.

A fundamental aspect of this ecosystem is the connection and interoperability between control layers and active assets (devices). Supportive robots that perform repetitive and low-value-added tasks, such as sample transportation, serve as crucial enablers in enhancing laboratory workflows.

Despite their importance, there exists a gap for a widely accepted, harmonized framework that addresses these needs. The Laboratory Automation Plug and Play (LAPP) concept, which is the focus of this thesis, proposes a semantic schema for service descriptions by hierarchically decomposing the laboratory workflows. These range from high-level *service* and *assay* levels through outcome-oriented *tasks* and *subtasks* to low-level *motion sequences*, *motion primitives*, and *actuator primitives*.

I develop an asset-centric information model based on the Digital Twin (DT) principle, aimed at enabling teaching-free integration of supportive laboratory robots. A key element of this model is the representation of robot positions, which act as endpoints for motion primitives. The resultant motion sequences constitute the subtasks and tasks involved in labware manipulation.

I introduce a hierarchical system architecture model built on contemporary Industry 4.0 principles. This model is mapped to the hierarchical workflow representation levels. In doing so, I delineate the harmonized roles of each layer and component within the ecosystem. I assign each layer an appropriate level of abstraction and clarify the nature of communication between components.

Finally, I present two feasibility studies that illustrate key components of the LAPP framework. The first study involved the development of an academic prototype employing a research-oriented Mobile Manipulator robot (MoMa) platform. Utilizing the Robot Operating System (ROS) in conjunction with the Standardisation in Laboratory Automation Consortium (SiLA) interoperability protocol, this solution implemented the *LabwareTransfer* feature.

Building on insights from this preliminary study, in collaboration with an industrial system integrator, we developed a pilot implementation within an actual laboratory environment. This implementation incorporated a widely-accepted scheduler layer alongside the aforementioned *LabwareTransferController* SiLA feature definition, integrated with the harmonized LAPP architecture.

The LAPP framework offers a comprehensive architecture model that amalgamates elements from established standards. Moreover, critical aspects of this framework have been integrated into the extensions of the global laboratory automation standard framework of SiLA.

KIVONAT

A támogató technológiák, mint például a mintaszállító robotok, széleskörűen elérhetők a laboratóriumi automatizálás területén. Az alkalmazási területek a kémia, az anyagtudományok, a biológia és a biotechnológiai területek tudományos kutatásától a gyógyszeripar Research and Development (R&D), Quality Control (QC) és folyamatszabályozási területein át egészen az egészségügyi és élelmiszeripari alkalmazásokig terjednek. A laboratóriumi automatizálás kulcsszerepet játszik az analitikai módszerek és kísérletek hatékonyságának és reprodukálhatóságának növelésében.

Az irányítási rétegek és az ökoszisztéma aktív eszközei közötti kapcsolat és interoperabilitás alapvető követelmény. Az ismétlődő és alacsony hozzáadott értékű tevékenységeket, például a mintaszállítást megvalósító támogató robotok elengedhetetlenek.

Jelenleg hiányzik egy széles körben elfogadott, átfogó és harmonizált keretrendszer. A jelen dolgozat tárgyát képező Laboratory Automation Plug and Play (LAPP) koncepció a funkcionális leírások szemantikai sémáját határozza meg a laboratóriumi munkafolyamatok hierarchikus alábontásával. Ezek a magas szintű *service* és *assay* szintektől az eredményorientált *task* és *subtask* szinteken át az alacsony szintű *motion sequence*, *motion primitive* és *actuator primitive*-ig terjednek.

Egy Digital Twin (DT) elven alapuló eszközközpontú információs modellt állítok, amely elősegíti a támogató laboratóriumi robotok tanítás nélküli integrációját. Ennek egyik döntő szempontja a robot pozíciók reprezentációja, amelyek végpontként szolgálnak a mozgásprimitívek számára. Az így létrejövő mozgássorok (motion sequence-ek) alkotják a laboratóriumi hordozók (labware-ek) manipulációjának részfeladatait (subtask-jait) és feladatait (task-jait).

Hierarchikus rendszerarchitektúra modellt definiálok az Ipar 4.0 ökoszisztémák modern elvei alapján. Ennek a modellnek a szintjeit leképezem a hierarchikus munkafolyamat reprezentáció szintjeire. Ezzel meghatározom az egyes rétegek és összetevők összehangolt szerepét az ökoszisztémában. Minden réteghez hozzárendelem az absztrakció megfelelő szintjét, és meghatározom a komponensek közötti kommunikáció jellegét.

Végül bemutatok két megvalósíthatósági tanulmányt, amelyek demonstrálják a LAPP keretrendszer kulcsfontosságú szempontjait. Elsőként egy akadémiai prototípus-fejlesztési projekt egy kutatás-orientált Mobile Manipulator robot (MoMa) platform használatával. A Robot Operating System (ROS) alapján és a Standardisation in Laboratory Automation Consortium (SiLA) interoperabilitási protokollt használva ez a megoldás megvalósította a laboratóriumi hordozók transzportjára irányuló *LabwareTransfer* funkciót.

Az előzetes tanulmány eredményei alapján egy ipari rendszerintegrátorral közösen létrehoztunk egy pilot megvalósítást valós laboratóriumi környezetben. Egy széles körben használt ütemező (scheduler) réteget és a korábban említett *LabwareTransferController* SiLA funkcionális definíciót (feature definition), valamint a harmonizált LAPP architektúrát használtuk.

A LAPP keretrendszer általános architektúra-modellt biztosít a széles körben elfogadott szabványok elemeinek kombinálásával. Ezenkívül a keretrendszer kulcsfontosságú szempontjait beépítették a SiLA globális laboratóriumi automatizálási szabvány keretrendszerének kiterjesztéseibe.

DECLARATION

Undersigned, Ádám Wolf, hereby I state that this Ph.D. thesis is my own work, wherein I only used the sources listed in the references. All parts taken from other works, either as word-for-word citations or rewritten, keeping the original meaning, have been unambiguously marked, and a reference to the source was included.

NYILATKOZAT

Alulírott Ádám Wolf kijelentem, hogy ezt a doktori értekezést önállóan készítettem, és abban csak az irodalmi hivatkozások listájában szereplő forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, November 4, 2025

.....

Ádám Wolf

Contents

Acknowledgment

I would like to express my gratitude towards my supervisors, Károly Széll and Péter Galambos. They led me through the long journey of conducting the research work, connected to my industrial activities, and helped me strengthen the scientific background of my work.

This work would not have been possible without Stefan Romeder-Finger's visionary decision to commit to the topic of laboratory automation and robotics. Within a pharmaceutical sciences organization, he realized the importance of this interdisciplinary domain early on and provided me with great support to kickstart the journey. Together with Thomas Gatternig, Jean-Baptiste Farcet and Paul Majneri the team was an excellent incubator of innovative ideas.

David Wolton, Julien Janda, and Josef Trapl helped to embrace the synergies with the manufacturing domain, which was an essential catalyst for creating the Laboratory Automation Plug and Play framework.

Guiding the journey within the pharmaceutical sciences domain, Patricia Wildberger and Christoph Pistek helped pave the road toward the implementation of the prototype and pilot projects. The execution of these would not have been possible without academic and industrial partners. Panna Zsoldos from the Óbuda University played a key role in the prototype implementation, while the Omron and EngRoTec teams conducted the industrial pilot.

The Standardisation in Laboratory Automation Consortium provided an excellent platform for manifesting the concepts as industrial standards. I would like to specifically thank Patrick Courtney for his mentorship.

Finally, my biggest thank-you goes to my friends and family for supporting me throughout this and all preceding journeys that led to the creation of this thesis.

I would like to dedicate this work to my fiancée, Veronika, whose dedication and perseverance in tackling challenging endeavors in her professional life have been a constant source of inspiration. Her deep enthusiasm for a wide range of topics has not only inspired me but also provided valuable knowledge from which I've directly learned. Most importantly, she has been my unwavering support, encouraging me through every high and low of the creative process.

Structure of the Thesis

As the table below illustrates, the thesis is organized into six distinct chapters. Chapter 1 provides an overview of the context, historical background, and motivations of the thesis work. Chapters 2 through 4 present the new scientific results, each comprising respective sections about the subject's background, state-of-the-art, and the motivation behind the new advancements. Each main chapter closes with the formal thesis statements summarizing the new scientific results. Chapter 5 presents two feasibility studies that address the presented concepts. Finally, Part 6. summarizes the contributions of the Thesis and outlines potential future research directions. The table also displays the corresponding publications, respectively. Note, that the preliminary review [WA1] and preliminary study on device integration [WA2] are cited by the respective thesis points. Similarly, the *Prototype* [WA6] and *Industrial pilot* [WA7] implementations are each covering both TH1 and TH2.

1 Preliminaries		
Theses		
2 Semantics [WA1, WA3, WA5, WA6, WA7]	3 Digital twin [WA3, WA4]	4 System architecture [WA2, WA3, WA5, WA6, WA7]
5 Implementations		
Prototype	Industrial pilot	
6 Summary		
Future work		

TABLE 1
STRUCTURE OF THE DISSERTATION

Notations and Symbols

AAS Asset Administration Shell
ABC-iRob Antal Bejczy Center for Intelligent Robotics
AMR Autonomous Mobile Robot
APC Advanced Process Control
API Application Programming Interface
AR Augmented Reality
BioSASH BioLAGO SiLA 2 AniML Serial Hackathon
BPMN Business Process Model and Notation
cobot collaborative robot
CS Coordinate System
DCS Distributed Control System
DCU Device-level Control Unit
DT Digital Twin
EC Embedded Controller
EDL Experiment Design Language
ELN Electronic Lab Notebook
ERP Enterprise Resource Planning
FM Fiducial Marker
FTE Full-time Equivalent
GMP Good Manufacturing Practice
HMI Human-Machine Interface
HPLC High-performance Liquid Chromatography
HTP High Throughput
IO Input-Output
IoT Internet of Things
IPA Fraunhofer Institute for Manufacturing Engineering and Automation
IPC Industrial PC
ISA International Society of Automation
ISPE International Society for Pharmaceutical Engineering
JTIO Joint trajectories and IO
LabOP Laboratory Open Protocol
LADS Laboratory and Analytical Device Standard
LAPP Laboratory Automation Plug and Play
LES Laboratory Execution System
LIMS Laboratory Information Management System
LoA Level of Autonomy
LPL Laboratory Process Language

LRP Low-level Robot Program
ME Motion Element
MECE Mutually Exclusive and Collectively Exhaustive
MES Manufacturing Execution Systems
MoMa Mobile Manipulator robot
MRP Modular Robot Program
MS Mass Spectrometry
MTP microtiter plate
OPC-UA Open Platform Communications - Unified Architecture
PAT Process Analytical Technology
PC Private Computer
PID Proportional – Integral – Derivative
PLC Programmable Logic Controller
PoI Point of Interest
PWM Pulse Width Modulation
QC Quality Control
R&D Research and Development
RAM Reference Architecture Model
RAMI4.0 Reference Architecture Model Industrie 4.0
RAR Robotic Activity Representation
REST Representational State Transfer
RoI Return on Investment
ROS Robot Operating System
RSL Relative Spatial Locations
SBOL Synthetic Biology Open Language
SCADA Supervisory Control and Data Acquisition
SCARA Selective Compliance Assembly Robot Arm
SiLA Standardisation in Laboratory Automation Consortium
SLL Symbolic Lab Language
SME Subject Matter Expert
SoA Service-oriented Architecture
SOP Standard Operating Procedure
SP Service Protocol
SRWG SiLA Robotics Working Group
TAT Turnaround Time
TCP Tool Center Point
TLA Total Laboratory Automation
TRL Technology Readiness Level
UI User Interface

UML Unified Modeling Language
URDF Unified Robot Description Format
USC Universal SiLA client
VDMA German Mechanical Engineering Industry Association
WBS Work Breakdown Structure
XDL Chemical Description Language
ZVEI Zentralverband Elektrotechnik- und Elektronikindustrie

List of Figures

List of Tables

Part 1

PRELIMINARIES

Automation and the corresponding supportive technologies are omnipresent across the life science laboratory landscape. The present thesis focuses on creating an integration framework in this context. By unifying the semantics, the information models, and the system architecture, the aim is to facilitate the interoperability of the supportive subsystems with the primary automation ecosystem. To set the scene, let us first consider the context of the application area, and then get an overview of state-of-the-art approaches.

1.1 Historical background

The history of pharmacy dates back to ancient civilizations, where the preparation of medicinal substances was intertwined with religious and mystical practices [7]. Early pharmacists, often priests or shamans, used natural ingredients such as herbs, minerals, and animal products to create remedies. The profession began to formalize in ancient Greece and Rome, with figures such as Hippocrates and Galen laying the foundations for modern medical and pharmaceutical practices [8]. During the Middle Ages, Islamic scholars preserved and expanded pharmaceutical knowledge, which later spread to Europe through translations of their works [9]. The Renaissance and Enlightenment periods saw the establishment of apothecaries (as shown in Figure 1.1), which evolved into more structured and regulated pharmacies [10]. The nineteenth and twentieth centuries brought significant advances with the development of synthetic drugs, the standardization of pharmaceutical education, and the introduction of legislation to ensure drug safety and efficacy [11]. In recent decades, technological advances such as laboratory automation [12] and biotechnology [13] have revolutionized the field, making modern pharmacy a sophisticated and integral part of healthcare. Modern day pharmacy relies on a highly regulated global industry, where advanced technologies, stringent quality control measures, and extensive research and development ensure the safety, efficacy, and accessibility of medications worldwide.

1.2 Modern pharmacy

The broad landscape of life science laboratories can be illustrated by looking at a cross-section of the modern pharmaceutical industry. Let us consider the life cycle of a drug

Fig. 1.1. Apothecary shop. Nuremberg : M. Endter, 1716 Credit: Wellcome Collection, CC BY 4.0

product. In general, drug development starts with early-stage basic research, which is usually conducted in an academic or clinical context. Drug candidates are then screened, and a subset proceeds to preclinical and clinical development. Before the new drug can enter production, approval from the authorities is required [14]. Throughout the process development cycle, several scale-up stages can be observed, from laboratory- through pilot- all the way to manufacturing scale. Progressively, each stage involves more and more regulations, ultimately reaching a Good Manufacturing Practice (GMP)-ready stage.

In recent decades, the boundary between laboratory and parts of the production ecosystem has become blurry. The complexity of chemical and biological manufacturing processes has increased to the level requiring advanced Process Analytical Technologies (PATs).

1.2.1 Pharmaceutical manufacturing

Advanced Process Control (APC) systems are gaining importance in pharmaceutical manufacturing. In most cases, they are contained, in the sense that the flow of the materials (mostly liquids) takes place in a closed system made out of tubes, tanks, pumps, and valves, which connect the unit operations with each other. These processes can be conducted in a continuous fashion or in batch operation. In many cases (especially in biotechnology), sample taking, e.g., for the purposes of at-line or offline analytics, needs to take place via aseptic (e.g., welded or sealed) connections, in order to prevent (cross-) contamination. As soon as a sample is drawn out of the (otherwise contained) system, its sterility cannot be guaranteed anymore, and the sample cannot be fed back into the system.

In-line sensors can be directly introduced into the flow, which provides real-time data to the APC systems. Besides that, at-line PATs requires taking a sample from the specific unit operations and feeding the sample into specialized analytical equipment. At-line analytics can range from simple single-device analytics, such as High-performance Liquid

Chromatography (HPLC), Mass Spectrometry (MS) or plate reading, to multi-step assays performed on a set of devices. This method provides data in a delayed, semi-real-time fashion. Finally, in an offline analytics scenario, the sample is taken away from the process to a dedicated laboratory, where typically longer and more complicated assays can be performed.

At-line and offline analytical assays are often plate-based, which means that an array of samples is pipetted to microplates, which are then loaded into subsequent analytical instruments, e.g., liquid handling robots and plate readers.

The data resulting from analytics can be used by the APC system to get an update on the state of the process. This value can be used for various control approaches, ranging from linear Proportional – Integral – Derivative (PID) and state-based control through nonlinear and model-predictive methods to machine learning systems. As in every control system, the circuit closes by *acting* on the system. This can be done by various actuators, ranging from flow control by valves and pumps through thermal management with heating or cooling to mechanical actuation, such as stirring or shaking. Both in the case of PATs and in the case of actuation, automation technologies, and robotics play a crucial role, representing both the system's "senses" and "muscles".

Besides APC, analytical technologies are also crucial for Quality Control (QC) purposes, where a variety of predefined and regulated tests must be performed on samples from various stages of production. In this regard, the trend of moving the analytics closer to the process (striving for in-line or at-line techniques, where possible) can be observed.

1.2.2 Pharmaceutical Research and Development

The efficiency of pharmaceutical R&D has been declining since 1950 [15]. This is due to an array of factors, including the saturation of the market, regulators raising the requirements towards new drugs, and inefficient allocation of resources. This tendency shows a similar - but reverse - curve to the well-known Moore's law. Moore's law explains the exponential evolution in electronics by stating that the number of transistors in an integrated circuit roughly doubles every two years. In contrast, Eroom's law states that the number of drugs per billion US$ is declining logarithmically [16]. On the other hand, also a reproducibility crisis is observable in R&D, which means that approximately two thirds of academic research is not repeated by the very few peers who decide to take up the unrewarding task of reviewing and repeating experiments [17, 18, 19].

1.2.3 Laboratory automation

Automation of life science laboratories is motivated by various factors, depending on the application. In High Throughput (HTP) screening or QC scenarios, for example, repetitive tasks must be performed in a structured environment with relatively low process variability. In contrast, a more dynamic R&D setup may pose a less structured and less constant environment.

The former scenario can be addressed by traditional means of automation, such as customized and hard-programmed robotic cells and integrated automation lines [20, 21]. These solutions resemble special-purpose equipment in the automotive industry and other

production environments, where such custom-made cells serve a specific purpose, e.g., to perform a certain assembly sequence or end-of-line testing.

If a process is highly repetitive, a high throughput can be achieved by adapting the components to that specific set of tasks [22, 23]. This means that the whole environment can be optimized for automation by de-prioritizing human ergonomic factors and prioritizing automated material and information flow. In the extreme case of a so-called lights-out operation, no humans are present in the space, be it a warehouse, a factory or a laboratory [24]. In laboratory automation, however, this vision cannot be achieved with the current state of technology in most cases. The most suitable venue for the implementation of this would be in a facility where routine tests are conducted, for example, in a healthcare diagnostic laboratory or in QC [25, 26]. The newly emerging concept of *cloud labs* also leverages these approaches with the aim of democratizing resources, just as was achieved by cloud computing.

These setups ensure a high throughput and repeatability, but lack the flexibility necessary for dynamic workflows. For distinct tasks, such as liquid handling and labware transportation, modular and easily reconfigurable solutions already exist [27, 28, 29]. To perform more complex activities flexibly, such as device- and tool interactions, novel robotics approaches must be adapted. These solutions include increasingly advanced perception, cognition, knowledge representation, and information sharing capabilities [30].

Diverse laboratory requirements have resulted in varied types and degrees of laboratory automation. Since there is no single definition for laboratory automation classification, two viewpoints can be mentioned to categorize automation levels. First is a generally accepted classification method, exploring the complexity of instrument integration, scaling from no automation through partial laboratory automation to Total Laboratory Automation (TLA), as a spectrum [31]. In the *no automation* scenario, all tasks and processes are conducted manually, whereas in *TLA*, various analyzers are physically integrated into a modular system resembling an assembly line. *Partial automation* exists between these extremes, involving automated tasks, subtasks, or parts of sample handling processes. However, even in partially automated labs, manual transportation between processing stations still requires the involvement of laboratory workers [32].

An alternative classification approach can be drawn from the realm of autonomous vehicles, specifically the standard SAE J3016 [33] which has been created for straightforward classification of automation levels, commonly referred to as *Level of Autonomy (LoA)*. This six-level scale has been adopted by other fields, including surgical robotics [34]. Based on this trend, a comparable six-level scale has been proposed for biological laboratories [35].

Assessing the financial aspects of laboratory automation presents a challenge, which deviates from the traditional Return on Investment (RoI) calculation exercises that are frequently used in industrial settings. The complexity and the often low Technology Readiness Level (TRL) of the necessary technical solutions increase the initial cost, both financially and in terms of resources. Laboratory automation solutions also do not directly replace the human workforce, but rather augment and support it. Hence, calculating the Full-time Equivalent (FTE) reduction is not trivial. However, with automation, the overall utilization of the available laboratory equipment can be drastically increased, by enabling operations overnight or during the weekend. Striving for 24/7 operation, the Turnaround Times (TATs) can be drastically decreased.

Additional benefits of automation and robotics can increase the connectivity of laboratory processes to the digital infrastructure. Besides enabling APC and QC as described above, this tight coupling of the physical and the digital worlds can also enable data acquisition on a higher level. Injecting, contextualizing, and storing this increased amount of data in a suitable manner and coupling it with advanced analysis techniques facilitates the monitoring of performance and decision making.

1.3 Robotics in laboratory automation

Robots are used for a wide range of tasks in life science laboratories [36, 37, 38] [WA1] [WA1]. In laboratories where higher flexibility is needed, a hybrid human-robot operation is desired. Many tasks, for example, measuring out a certain amount of powder from a container, require a level of dexterity that is not yet achievable by conventional robots. Although many laboratory devices are already automated in a standalone manner, most of them are not optimized for being integrated into an overlaying automation system [39]. In most cases, they are still designed mainly for human operation with operator interfaces such as touch screens and keyboards [40]. This means that human workforce is still needed, e.g., to load the samples into the system and transport them from one device to the next.

In general, life science automation systems can be differentiated based on the following factors [36]:

- Local distribution of the automation devices (centralized or decentralized)
- Flexibility of the automation structure (open or closed)

According to the definition by Fleischer et al., the biggest challenge for system integration lies in decentralized/open systems. Contrary to most traditional industrial applications, where the processes are not changing for a longer period (usually years), in life science laboratories a supportive robot has to cover a range of functions. When acting as a central system integrator, the robot serves as a transport element. For this, either the devices must be adapted, or the robot has to mimic how a human would operate each device. On the other hand, Fleischer et al. distinguish a flexible robot by the fact that besides transportation it also performs other laboratory tasks, such as sample manipulation [41].

According to Fleischer et al. [36], laboratory robotics solutions can be designed in two general configurations:

- Fixed-position robots surrounded by all devices, tools, labware and consumables.
- Mobile manipulators, which can approach different stations while transporting and manipulating the samples. This approach implements an open automation system.

1.3.1 Material handling

In laboratory automation systems, supportive robots and peripherals can play a crucial role in material flow between individual stations. Components often need to be operated in a mixed mode by both humans and robots. Therefore, the applied robotic equipment

must meet certain safety requirements, which are, however, not fully defined yet. In contrast to traditional industrial applications, where robots were separated from humans by physical barriers (cages), cobots are capable and allowed to work in a shared human-robot workspace. Collaborative robots must avoid collisions with humans or at least minimize the collision forces, as specified in the technical specification ISO/TS 15066:2016 [42]. To achieve this, multiple technical solutions exist, ranging from torque sensing and limitation through sensitive robot skins to monitoring the environment with external sensors.

Benchtop robot arms, stationary or rail-mounted, are now available from multiple integrator companies of laboratory automation systems. These applications usually focus on handling standardized sample carriers, such as ANSI/SLAS-conform[1] microplates. The scheduler software controls both laboratory devices and transportation robots, especially starting procedures and monitoring their states. However, such systems require comprehensive integration efforts, in that the placement of the devices around the robots or the tracks has to be carefully designed. In most cases, custom frames are needed instead of the conventional laboratory benches. With this approach, sets of devices can be integrated into an automated island, where the robot serves as a transportation backbone. However, this approach introduces spatial constraints, hence highly limiting the flexibility [WA1].

1.3.2 Flexible robotization

Mobile Manipulator robots (MoMas) consist of a mobile base (Autonomous Mobile Robot (AMR)), one or more robot arms, and usually a number of sensors, including laser scanners and cameras [43, 44, 45]. These units are capable of navigating to various stations in a facility, picking up objects, such as packages, workpieces, or samples, transporting them to the next station, and eventually loading them in a device or machine for further processing or storage. Being able to cover a larger area than a fixed-base or rail-mounted robot, mobile manipulators can increase the flexibility of automated laboratory systems. Their range can cover multiple laboratories across different floors of the same building. For this, the interface with automated doors and elevators must be implemented [46].

Fleischer et al. [36] consider a MoMa as an integrated robot solution in that it can fulfill both sample transportation and manipulation in an open automation system. To enable the robot to interact with laboratory devices, the interfaces must be defined. This includes both the communication and control interfaces to be configured and the movements of the robot to be programmed. The latter component calls for an appropriate representation of the movements.

When it comes to fulfilling handling tasks with any type of robot, precision and accuracy are key considerations. As a reference, it has to be mentioned that robots in laboratory automation are mainly used for automating tasks that were traditionally conducted by humans. As such, the required precision is hard to define. As a reference, the pipetting and bench-top handling robots can be considered. A popular example of the former, the TECAN Freedom Evo, has 0.5 mm specified for the robotic manipulator arm, whereas the PreciseFlex Selective Compliance Assembly Robot Arm (SCARA) arm is given a 0.09 mm value. The precision of an AMR on its own lies in the cm range, which has to be

[1] Meets the Standards ANSI/SLAS 1-2004 through ANSI/SLAS 4-2004

improved by the means of additional position detection methods. Overall, a precision in position of around 1 mm and in angle around 1 deg is desired.

Fiducial markers play a crucial role in many of the state-of-the-art MoMa applications. Principally, a fiducial marker is a two-dimensional (usually binary) pattern that can be printed and sticked to various surfaces to serve as an optical marker for pose estimation. A computer vision algorithm is used to detect the pattern on a calibrated camera's image and calculate its corresponding position and orientation in relation to the camera coordinate system (frame). One of the traditional applications was Augmented Reality (AR), where virtual objects can be added to a camera-captured scenery dynamically. In these applications, fiducial markers serve as a reference between the physical and the virtual environment in the form of coordinate systems. Garrido-Jurado et al. propose a method for the generation and detection of fiducial markers called ArUco [47]. Besides AR, they also mention robot localization as one of the application fields for the technology.

In the discussed applications, fiducial markers are used as user coordinate systems in which the robot motions must be composed. As such, a user can define the grasping pose in relation to the marker by moving the robot arm to the desired configuration either by hand or by jogging.

Considering the basic functionality, using an AMR in combination with fixed base robots can serve as an alternative to a MoMa. The two different approaches both have their advantages and disadvantages and ultimately, the specific use case determines which one proves to be the optimal solution. On the one hand, when a higher throughput has to be achieved (e.g., in manufacturing or logistics), setups with conveyor belts and/or fixed base robots must be considered. MoMas might not reach the speed of humans, but they can get closer to their level of flexibility than fixed base robots specialized for one certain task. Although a single robot arm might be lower-priced as a complex MoMa, the latter can render multiple stationary units superfluous by fulfilling multiple handling and tending tasks at once. Being able to work 24/7 gives these systems an advantage in comparison to human-based solutions. See Table 1.1 for a comparison.

TABLE 1.1
COMPARISON OF FIXED BASE AND MOBILE ROBOTS WITH HUMAN WORKFORCE

	fixed base	MoMa	Human
Throughput	High	Low	Middle
Availability	High	Middle	Low
Flexibility	Low	High	High

Multiple system integrator companies have already begun to provide MoMas besides the usual stationary or rail-mounted SCARAs as part of their sample transportation solutions. Many of these feature the same type of PreciseFlex robot arm, which has become a standard for benchtop laboratory robotics [48]. As such, the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) has launched KEVIN, which is based on a Care-o-bot 4 mobile platform, a mounted PreciseFlex arm and a vision system, which is capable of localizing fiducial markers [49]. Biosero has provided a very similar solution [28]. They use the same SCARA, but in this case it is mounted on an OMRON LD90 mobile base, alongside with a different vision system for detecting a different kind of fiducial marker. The solution of UniteLabs and Astech Projects only slightly differs from the above two in the sense that it is based on a mobile platform with a larger footprint and that in-

stead of a SCARA it features a 6-axis arm from Universal Robots [50]. In addition to that, the fiducial-based pose detection principle is the same. Moving towards industrial robots, KUKA's MoMa named KMR IIWA must also be mentioned [51]. This specific robot was used by Burger et al. for conducting photocatalysis experiments within a ten-dimensional space [52]. It can be concluded that the blueprint of equipping a mobile platform with at least one robot arm and a vision system capable of fiducial-based pose detection is already well established and widespread in the laboratory automation landscape. The use of MoMas also goes beyond the walls of the laboratory when it comes to the pharmaceutical industry. For the Marvin Project of the PM Group and the University College Dublin, an iiwa unit was used for environmental monitoring purposes [53, 54].

Taking over a wide variety of tasks from humans is also beneficial for minimizing on-site presence during the times of a pandemic, such as COVID. Besides fulfilling transportation and manipulation tasks, MoMas can utilize their wide range of sensors and actuators, which opens up additional possibilities. As such, an AMR or quadruped (four-legged mobile robot) can also be used for remotely monitoring an automated system in a 24/7 operation [WAO3]. Without having to travel to the facility, stand-by personnel can remotely control the robot to navigate to the place where the error occurred. With the robots on board camera and additional sensors, the cause of the error can be evaluated, and, in certain cases, telemanipulation can be used to resolve the error. As a simple example, a freezer door left open can be mentioned, which can easily be closed with a push. Similarly, in the case of the stationary sample-handling robots, a misplaced plate can be pushed back to the handover nest, that is if no spillage occurred.

1.4 Standardized interoperability

It can be seen that integrating standalone laboratory devices into end-to-end or hybrid automation ecosystems is necessary to achieve the desired efficiency. Besides implementing the automated flow of materials (e.g., via robotic solutions), it is essential to enable the interoperability of these units by means of an overarching orchestration and scheduling system. The control of laboratory devices is possible via specific Application Programming Interfaces (APIs) in the form of triggering commands that represent the functionalities and steps of an experimental sequence (assay).

Most vendor-provided APIs of laboratory devices is proprietary, that is, not compliant with any industrial standard. System integrators usually create drivers in the form of wrappers around these APIs, to enable the integration of the device with their scheduler. However, integrator-proprietary communication protocols limit the compatibility to clients (e.g., schedulers) of the specific integrator. In contrast, implementing a native API that is compliant with an open and documented industrial standard ensures compatibility with a wider range of clients. Alternatively, instead of an integrator-proprietary driver around the device API, a standardized one would also serve the same purpose. The development and incorporation of a standardized interface could eliminate the need for repetitive work in this area.

A solution for this problem was developed by the Standardisation in Laboratory Automation Consortium (SiLA) [55] [56]. The organization consists of software suppliers, system integrators, and users, such as pharmaceutical and biotech companies, and aca-

demic institutes. SiLA has created an emerging standard by the name SiLA2, which aims to define the interface between laboratory instruments and the automation software system [57].

In the SiLA-ecosystem, functionalities and capabilities of lab devices are represented as *feature definitions* in XML format. Although there are conventions for creating feature definitions, there are still a few possible ways to define them for each type of device, depending on the use case and laboratory setting. This results in different definitions among different vendors. Examples of this are the feature definitions for MoMas from the Fraunhofer Institute for the Kevin robot [29] or from Astech Projects [58].

Every feature definition can consist of commands and properties for basic or more complex and high-level tasks. Properties are usually responsible for returning read-only values through SiLA calls, while commands usually start/stop a process, set values, and other functions that require some modification.

The Laboratory and Analytical Device Standard (LADS), based on the Open Platform Communications - Unified Architecture (OPC-UA) framework, mainly targets the same problem statement as SiLA, namely the integration of laboratory equipment with control systems [59]. The OPC-UA base layer, originating from industrial automation and the process industries, features a wide variety of off-the-shelf capabilities for production equipment, e.g., for state-based control, modularization, and inheritance between module descriptions or relative spatial position representations for robots. LADS, as a companion specification under the control of the German lab instrument trade association Spectaris, adds a domain-specific layer to OPC-UA, focusing on the specifics of laboratory equipment and the control systems. Laboratory devices are represented in terms of functional and physical modules, which facilitate basic automation, orchestration, and service & asset management. The specification [60] having been released only in November 2023, its ubiquity is limited at the time of writing this thesis.

1.5 Research goals

The primary objective of this research is to establish a standardized interoperability framework for laboratory automation, specifically targeting the integration of various automation units to form end-to-end comprehensive workflows. This framework aims to enhance the automation platforms used in modern laboratories by focusing on supportive lab robotics. The specific research goals are as follows.

1. **Assessment of laboratory workflows and definition of semantic representation formats**
    - Evaluate the existing workflows and operational methods in contemporary automated and non-automated laboratories.
    - Identify suitable activities that can be automated and augmented using supportive robotics.
    - Develop a harmonized semantic representation format for different levels of granularity in laboratory activities.
    - Identify appropriate service providers, device components, and actors within the system to perform each type of activity, with a focus on supportive robotics.

2. **Development of information models for integration**
    - Create information models that facilitate the interoperability and plug-and-play integration of both active and passive components.
    - Ensure that these models accommodate the specific needs of robotization, including geometric information and positional data for robotic operations.
3. **Comprehensive system architecture model**
    - Define a system architecture model that delineates various levels and canonical sequences for automated and robotized laboratory systems.
    - Assign levels of activity representation to corresponding levels of the control architecture to enable seamless plug-and-play compatibility and integration of automated laboratory devices.

By achieving these goals, the research aims to provide a robust and flexible framework that supports the integration of various laboratory components, thus advancing the capabilities and efficiency of automated laboratory systems.

Methods

As described in 1, laboratory automation has a broad application area across academic research, healthcare, and the pharmaceutical industry. The latter serves as a good textbook example for getting an overview about the different aspects of life science laboratory automation by viewing the life cycle of a drug product from its conception in early stage discovery, through the process and product development, licensing, production and QC.

The breadth of the application area entails a wide variety in terms of the nature and complexity of the workflows that are to be automated. Process control and experiment orchestration can be named as the two main domains, which each have their own intricacies, but also their overlaps. In detail, I will discuss the system architectural aspects of these two domains in section 4.2.4.

In this dissertation, I present the Laboratory Automation Plug and Play (LAPP) framework for the integration of supportive robotics technologies into laboratory automation ecosystems. I define the scope and focus of LAPP to orchestrated analytical assays that are labware-based (in most cases plate-based). However, I will also discuss the potential applicability of the framework in combination with contained APC systems.

Using a multidimensional representation, I characterize the elements of the laboratory automation system along three dimensions. For all of these aspects, I span a scale of abstraction and granularity.

1. **Process decomposition**: Layers of the workflow representation with the aim of manual or automated execution.
2. **Protocols**: Activity representation formats or languages.
3. **Technical layers**: Layers of the control architecture.

For this purpose, I structure the thesis as follows:

In chapter 2, I present the semantic decomposition of workflows in life science laboratories, based on the evaluation of manual and semi-automated tasks in pharmaceutical process development laboratories.

In chapter 3, I propose an information model and an operating model for aiding the integration of laboratory assets with automation systems and robots.

In chapter 4, I present a generic architecture model (LAPP-Reference Architecture Model (RAM)) for the integration and automation of equipment in life science laboratories.

After outlining the three conceptual aspects of LAPP, in chapter 5, I provide two examples of its implementation. 5.3 shows an experimental (prototype) implementation with a research-oriented MoMa platform, representing a TRL 4 [61] implementation, while section 5.5 discusses a TRL 5-6 implementation in a real-life laboratory environment, with the use of industrial grade components.

Part 2

SEMANTICS OF LABORATORY WORKFLOWS

2.1 Motivation

To automate labware-based analytical assays, the workflows must be decomposed in a hierarchical fashion. This way, the suitable units of activities can be assigned to specific actors (devices or humans) in the lab. Orchestrating such multi-level workflows requires the adequate representation of the capabilities of each actor, in the form of service descriptions. These service descriptions can then be matched to the requirements that a workflow description (recipe) encodes.

2.1.1 Workflow representation in laboratory automation

I will discuss two categories to give an overview of the state-of-the-art approaches to workflow representation in laboratory automation.

First, let us consider the commercial scene, where laboratory automation device vendors [62, 63, 64], system integrators [65, 66, 67, 68] and software companies [69, 70] provide their own scheduler and orchestrator software. These software usually come with a built-in workflow representation solution. However, these are almost always based on a proprietary format/language and are limited in functionality, logic, and breakdown possibilities. Most of these solutions feature some type of graphical representation similar to a flow chart.

The second category comprises academic initiatives that aim to create open-source and independent frameworks. One such example is the LARA suite [71, 72], which features its own process description framework. As an elemental part of the LARA suite, the PythonLab language denotes the laboratory process using the Python syntax. A workflow graph parses this representation and passes it to the scheduler, which defines the optimal order of execution. Finally, the orchestrator executes the schedule and takes care of the communication with the devices.

Another example of an independent laboratory process representation protocol is Laboratory Open Protocol (LabOP) [73]. The objectives of the LabOP framework are multifaceted:

- **Inter-project Communication**: LabOP facilitates seamless transfer and comprehension of protocol representations across disparate projects.
- **Reproducibility**: The framework is tailored to ensure that different laboratories, even those with varied equipment configurations, can reproduce these representations with high fidelity.
- **Clarity and Reusability**: LabOP's representations are both unambiguous and abstract, striking a balance to promote reusability across diverse biological contexts.
- **Broad Accessibility**: The framework is designed such that both automated systems and researchers can readily interpret and utilize the representations.

To ensure reliability and adaptability, LabOP leverages established technologies including the Unified Modeling Language (UML), Autoprotocol, and the Synthetic Biology Open Language (SBOL). Moreover, LabOP's remit extends beyond mere protocol representation; it encompasses execution records, resultant data, and the requirements of the execution framework itself. Notably, UML offers an agnostic approach to semantically model behavior, ensuring compatibility and coherence in various scientific domains.

LabOP is responsible for representing biological workflows, where the implementation of a laboratory primitive is considered a device-dependent black box. Lab robots are used in most cases for supportive actions (e.g., labware transportation), which are not represented in the biological protocols. As such, neither does the Autoprotocol library include a specific primitive for labware transportation, nor does LabOP's extension. In addition to supporting actions, lab robots would also be suitable for performing certain meaningful activities, such as shaking the sample.

Another notable example of a standardized experimental description is the Chemical Description Language (XDL) [74]. Originally developed as part of the Chemputer project [75], XDL provides a comprehensive, machine-readable specification for chemical synthesis and related laboratory workflows. Its core objective is to enable unambiguous, platform-independent representation of experimental procedures, allowing them to be executed on any compatible automated system without modification. XDL captures all necessary experimental parameters, including reagents, reaction conditions, equipment configurations, and procedural steps, in a structured XML-based format. This level of formalization supports reproducibility, facilitates sharing of protocols between laboratories, and bridges the gap between human-readable experimental plans and machine-executable instructions.

To overcome the limitations of state-of-the-art laboratory workflow representation approaches, I introduce a new approach that draws inspiration from project management, industrial automation, and robotics to create a semantic workflow description framework. This should enable the agnostic representation of units of activities in laboratory automation by introducing a multi-level hierarchical decomposition framework. The appropriate levels of activities must be mapped to specific service descriptions using standardized communication protocols.

2.2 Preliminaries

In this section, I will provide an overview, outside of the laboratory automation domain, of different approaches for hierarchical workflow representation, and in particular for the representations for robotic activities.

2.2.1 Hierarchical workflow representation

Hierarchical decomposition is an effective way of comprehending the vertical of complex multilevel systems in different domains. First, I consider two examples from project management.

Work Breakdown Structure (WBS) [1, 2] is a project management tool that provides a deliverable-oriented breakdown of processes. This is to aid scoping, cost estimation, and work-package assignment. As I define Laboratory Automation Plug and Play (LAPP) hierarchical decomposition framework in Section 2.3, I adopt certain principles of WBS in the following fashion:

Similarities:

- Both structures serve the purpose of hierarchical decomposition of processes.
- Both definitions use Mutually Exclusive and Collectively Exhaustive (MECE) elements, in that they cover the corresponding activities unambiguously and explicitly.
- Their elements are outcome-oriented and agnostic toward implementation.
- WBS templates can be used to represent laboratory workflows with the LAPP notation.

Differences:

- A WBS breakdown usually has two to four levels, where the terminal element (the one that is not subdivided further) is a work package that a project member can meaningfully address.
- In LAPP, the depth, that is, the level of the terminal element, cannot be strictly defined. Each level can be considered to be the deepest necessary abstraction from the perspective of a certain control element. e.g., from the perspective of the lab automation scheduler low-level activities, such as robotic motion and actuator primitives, are not relevant.
- WBS does not provide activity definitions in either a symbolic/abstract or executable/geometric fashion.
- In the context of LAPP framework, I define Robotic Activity Representations (RARs). Their primary purpose is to describe robotic and automated laboratory device activities for execution at the appropriate control level within the laboratory automation system.

In the realm of hierarchical decomposition, a prime example can be found in issue trackers such as Jira [3, 4]. There, the following levels are distinguished:

TABLE 2.1
HIERARCHICAL DECOMPOSITION FRAMEWORKS IN PROJECT MANAGEMENT. WBS [1, 2] AND ISSUE TRACKERS (SUCH AS JIRA) [3, 4]

Level	**WBS**	**Issue trackers**
7		Initiative
6	Level 1	Epic
5	Level 2	Task/story
4	Level 3	Subtask
3		
2		
1		

TABLE 2.2
HIERARCHICAL DECOMPOSITION FRAMEWORKS IN SURGICAL OPERATIONS, ROBOT SURGERY, SERVICE ROBOTICS AND LABORATORY ROBOTICS.

Lvl	**Vedula [78]**	**D. Nagy [77]**	**Leidner [79]**	**Chu [76]**
7				
6	Procedure	Operation		
5	Task	Task	Action template	Task motion
4	Maneuver	Subtask		
3	Gesture	Surgeme	Operations	Motion element
2		Motion prmtv.	Geometric level	Motion step
1				

1. **Tasks**: At its core are *Tasks*. These denote individual work units that can be directly assigned to a designated person. They can exhibit various interrelationships, from dependencies to specific completion sequences.
2. **Subtasks**: A *Task* can be subdivided into *Subtasks*, which further detail the work.
3. **Stories**: On a broader scale, *Tasks* can be aggregated into *Stories*. A *Story* acts as an overarching task, encapsulating general information about a collection of *Tasks*.
4. **Epics**: As we progress further, we find *Epics*, which encapsulate the overarching objectives or themes of a project. It is common for *Stories* and *Tasks* within a single *Epic* to have interwoven dependencies.
5. **Initiatives**: At the zenith is *Initiatives*. These symbolize wider goals and might encompass multiple *Epics* from different teams, signifying a more extensive collective aim.

Table 2.1. puts the examples from the project management application area next to each other in terms of the levels of the breakdown. I use uniform level numbering throughout the dissertation, including tables 2.1, 2.2, 4.1 and in Section 2.4, where I outline the proposed LAPP hierarchical decomposition. This represents a loose mapping between the hierarchical layers of the different frameworks.

When it comes to robotic activities, it is important to define the different levels of representation. Chu [76] and D. Nagy et al. [77] both provide hierarchical decompositions in this regard, while Vedula et al. [78] analyze the structure of surgical activities from the perspective of a human surgeon. Although the field of application and the nomenclature are different, the corresponding levels can be identified, as presented in Table 2.2.

TABLE 2.3
TYPES OF CONVENTIONAL ROBOT MOVEMENTS

Name	Meaning	Characteristics
MoveJ	Joint move	Optimal joint-based movement with non-prescribed path
MoveL	Linear move	Prescribed linear path
MoveC	Circular move	Prescribed circular path
MoveP	Process move	Defined path and velocity (e.g., for welding or dispensing)

2.2.2 Robotic activity representation

Out of the layers of workflow decomposition (and of the system architecture), the low-level side is closest to the physical process. Some of these physical activities can be implemented through robotization. To do this, the decomposition must consider the specifics of the robot hardware, corresponding to a suitable level of granularity. In this context, different representation formats (also called protocols or languages) are used.

In its simplest form, a robot program is just a sequence of pre-taught movements and I/O operations, potentially with some basic logic operations, branching, and loops. Table 2.3 shows the various types of conventional robot movements (using Universal Robot's nomenclature [80]).

In more complex robot applications, simple movements might need to be grouped and organized for modular reusability. This enables a sequence of movements to be performed throughout the robot program multiple times, potentially on multiple different frames of reference.

In addition to industrial robotics, an increasing number of new application fields are arising. One of these is robotic surgery, which represents a significantly different set of requirements but utilizes many advantages of the fundamentals. As such, the robustness and reliability of robots is a key factor when it comes to teleoperated or (semi-)autonomous robotic surgery. The latter application means that during a robot-actuated surgery, that is, when the surgeon controls the robot's movements, the robot could take over some simple tasks to relieve the surgeon. To achieve this, tasks must be defined in a fully descriptive manner. As a basis of these representations, the analysis of how a human surgeon performs the task can be considered [78]. Nagy et al. propose a framework for surgical subtask automation based on the Da Vinci surgical robot, for which they provide a hierarchical structure to handle the different granularity levels of surgical motions [77].

Laboratory automation represents special needs towards robotic solutions concerning the complexity and variability of tasks [36]. Chu proposes the concept of Motion Element (ME) specifically for the automation of a complex sample preparation laboratory workflow [76]. They consider robot movements from one position to another as a *motion step* and group them in reusable units called *motion elements*.

The AI.Laboratory solution paradigm of Knobbe et al. represents a framework for robot-centered laboratory automation systems [81]. It incorporates the so-called Lab Skill System (LSS), a taxonomical collection of laboratory skills consisting of the basic and cell-specific subgroups.

Laboratory robots' capabilities are currently mostly limited to pick-and-place type transportation of labware by means of a parallel gripper. For the robot to perform these actions autonomously, a suitable knowledge representation is relatively simple. It must include the position of the object in relation to the robot, the object's gripping frame, and

the required gripper width. However, to broaden the scope of laboratory robot capabilities, new advances in robotics technologies must be utilized. To address the need for robots capable of working with a variety of objects in an unstructured human environment, service robotics approaches must be adapted.

As such, a very important aspect is how robots deal with complex objects, including tools, i.e., objects that are used in combination with the robot's own end effector to perform certain actions on other objects. This is how humans perform actions by hand with the use of hand-held tools. For that, humans use the dexterity of their end effector: a human hand with five fingers is incredibly flexible. In addition, cognition enables humans to perceive objects with a focus on their purpose. The corresponding capabilities and constraints are assigned to the objects intuitively. For robots to be able to achieve similar flexibility in performing actions with tools, suitable knowledge representations and the corresponding planning framework are needed, which utilizes these representations. Leidner addresses these needs in his early works and also in his dissertation [79, 82]. In chapter 3 I introduce the LAPP Digital Twin (DT) concept, which draws inspiration from Leidner's work for creating an asset-centric information representation framework for laboratory robots.

2.3 Hierarchical workflow decomposition in LAPP

In this section, I propose a process decomposition framework by outlining the layers of the workflow representation, with the aim of manual or automated execution. For this, I span a scale of abstraction and granularity. I define the dimensions of the hierarchical decomposition, and later in 4 also of the system architecture by using the same level numbering across the tables 2.4, 2.5, 2.6, 2.7 and 2.8. Levels 1-5 are also color-coded.

2.3.1 Process decomposition

Activities within a laboratory workflow can be hierarchically decomposed by describing the different levels of granularity. For this purpose, I utilize the principles of WBS, as discussed in 2.2.1. In the LAPP hierarchical decomposition each layer represents a level of abstraction, assuming that from that certain perspective, how the activity is implemented on the lower level is not important. As such, each representation layer is agnostic towards the lower (inferior) ones. This means that from the perspective of the scientific *Service* description it does not matter how the experiment itself is executed. Similarly, from the *Experiment* description, the device-level execution is abstracted away. Also, arriving at the robot-specific levels it can be deduced that a high-level *Task* description does not concern how the task is executed, instead, only the outcome is interesting. With these principles in mind I distinguish between the following levels of the process decomposition.

7) **Service** refers to the entirety of the laboratory's capabilities, e.g., high throughput and/or microscale services.

6) **Procedure** is an experiment, an assay, or a repetitive/continuous laboratory process.

5) T **Task** is an elemental action item that is carried out by a human or a certain device.

4) S **Subtask** is an intermediary layer that represents parts of a task, that accomplishes minor landmarks.

3) Q **Motion sequence** is defined for the specific case of robotics, whereby the robot performs a sequence of motions, for example, to approach a handover site.

2) P **Motion primitive** is an elemental motion of a robot or other mechanism.

1) A **Actuator primitive** is an output excerpted by a certain actuator, e.g., robot joint, pump, etc.

0) Physical process refers to the actual subject of the laboratory activity, e.g., a set of samples, buffers and/or reagents.

I present detailed examples for liquid handling in Section 2.3.2, and for robotics in Section 2.4.

2.3.2 Adaptation to liquid handling activities

To understand the hierarchical layers from a general (non-robot specific) activity's perspective, let us consider the example of a liquid handling task. As a crucial part of most assays, liquid handling means the transfer of certain volumes of liquids from a source container to a destination container. Also referred to as pipetting, liquid handling tasks can be carried out either manually by a human or automatically by a pipetting robot.

A more traditional and high-throughput-focused form of automated pipetting is the use of gantry-type liquid handlers, such as Tecan, Hamilton, or Beckman-Coulter machines. These units can be considered *Device* level entities that fulfill a certain T *Task* as a part of an *Experiment*. A pipetting robot executes a so-called liquid handling script, which contains S *Subtask* level commands, such as aspiration and dispensing, and in some cases also plate movements within the worktable of the machine. In addition, additional devices might be integrated into the liquid handler robot. In this case, they play a direct role in the script by performing certain subtasks either on the sample itself, such as shaking or incubation, or on the labware, such as washing. In this manner, the context of a liquid handler, which contains the integrated devices, can be considered as a robot cell, that is, a stand-alone unit. These, by themselves, can also be integrated into a bigger system (analogous to a production line), by interfacing with an overarching controller (or scheduler) and also being physically connected by the means of material transport. The same principle is repeated within their own context, where they act as an integration platform for their submodules and integrated devices. They implement both control (and information), and material flow. This means that the *Scheduler-Device* relation can manifest itself in a nested manner, to some extent, i.e., a S *Subtask* can ascend to become a T *Task* having its own S *Subtask*s. This shows that the boundaries of the hierarchical decomposition are fluid.

The S *Subtask* level is defined as an activity segment that accomplishes minor landmarks. As such, it can be shown why an Aspirate activity of a liquid handler is analogous to a GetLabware activity of a robot arm (see definition in section 2.4.3. In fact, a liquid handler robot by itself may perform pick-and-place tasks in the confines of its own worktable, with the help of its robotic manipulator. In general, a liquid handler, in essence being a robot, performs Q *motion sequence*s, such as approach sequences or gripping. Going one level deeper, some liquid handlers are able to define so-called *motion vectors* or points, which are analogous to P *Motion primitives*. At the deepest level, the axes of the gantry mechanics are controlled separately by the embedded controller to perform the

desired motions. This is analogous to the joint trajectory control of robot arms. To stay with liquid handling, the pump control can also be assigned to this level.

Table 2.9 maps liquid handling activities to the levels of hierarchical workflow decomposition, along with the supportive robotic activity of labware transfer, which I elaborate in Section 2.4.

2.4 Robotic activity representations in LAPP

2.4.1 Design principles

As already stated in Section 1, the LAPP framework's primary focus is on supportive laboratory robotics. Based on the hierarchical semantics framework, I outline a taxonomy, which provides a system for naming and organizing present and future lab robot capabilities. These representations must be robot-agnostic in the sense that multiple different robots (from different vendors) must be able to implement them. However, the definitions of these capabilities must include appropriate constraints for the planner (scheduler) to decide which robot can perform a certain task. This is a similar consideration to the binding between the semantic and the geometric description in Leidner's Action templates. Moreover, capability representations should align with the LAPP-DT concept (proposed in chapter 3, which necessitates that each robotic activity be depicted within the context of the corresponding laboratory asset, maintaining a relational perspective. This approach, akin to Leidner's object-oriented methodology, ensures comprehensive coverage. The representations must be agnostic towards the lower-level implementation by providing an appropriate level of abstraction and the use of symbolic definitions. These parameterizable representations will be used as commands during the execution of the workflow, each activity of the respective level being triggered by the superordinate control component.

I elaborate the taxonomy of laboratory robot activities on the conceptual level, agnostic to the implementation. The framework should fulfill the following requirements:

- Form a semantic description for multi-level activities of laboratory automation.
- Focus on laboratory robotics, including supportive activities, such as labware transport, and direct robotic implementations of primary activities, such as shaking.
- Categorize these activities according to their outcome and level of granularity.
- Nest low-level activities into high-level ones.

I will consequently use the same definitions as in chapter 3, including the hierarchical definition of the coordinate frames (positions), unless otherwise stated.

The application field for laboratory robots spans from labware transportation [29] to advanced manipulation and liquid handling [83]. The most ubiquitous of these are pick-and-place type labware transportation solutions, which are available off-the-shelf as market-ready products. I elaborate the LAPP RAR taxonomy by targeting this as a core capability. Additional supportive robot capabilities are still less established, and most of the solutions are still in the research phase. Thus, the variability in the technical approaches is high. I also provide a structure to categorize and incorporate these to-be-established capabilities in the taxonomy.

2.4.2 Survey

In June 2022, I conducted a targeted survey with the participation of seventeen experts in the field of laboratory automation. The majority of the participants identified themselves as *Users, laboratory experts* (seven), and *Robotics researchers* (six). In addition, the following groups were represented by one participant each: *Robot vendors*, *Lab software vendor*, *System integrator*, *Lab equipment vendor* and *Application engineer*. Twelve participants were active in the *Industry*, while five were in *Academia*. Fourteen were in *Research and development*, two in *Manufacturing*, and one in *Quality control*. The *Biologics* (six) and *Chemistry* (five) fields were represented by the majority, but *Software engineering*, *Bioinformatics*, *Biochemistry*, *Robotics*, *Mechanical engineering* and *Lab automation* were also marked. See the detailed analytics in the supplementary materials S1.

Participants were asked for which use cases they already use or provide robots for. *Gantry-type liquid handling* and *Benchtop robots for labware transportation* (eleven each) were the top use cases, while *Floor-level mobile manipulators for labware transportation* (six) and *Human-line liquid handling* (four) also proved to be widely used. *Benchtop collaborative assistance* (two), *Mobile collaborative assitance*, *Other labware transportation (e.g., drones)* and *Mobile robot for state monitoring* were also marked.

Following that, use cases were rated based on how likely the participants were to consider using or providing them in the future, based on their current state and future perspective. *Floor-level mobile manipulators for Labware transportation* proved to be the most popular (on average, 4,1 rating out of 5, where 5 is the most likely), then *Benchtop robots for labware transportation (e.g., PreciseFlex)* and *Benchtop collaborative assistance* followed (both with rating 4,0).

In the final section of the survey, participants were asked to rate and suggest laboratory robot activities in the form of commands. The supplementary table `Commands_survey-.xlsx` lists the commands in the order of decreasing average rating. It can be seen that the experts rated the commands the highest that belong under the *LabwareTransfer* task. This justifies the decision that LAPP is elaborated primarily to address this use case. The next subsection presents how the pick-and-place labware transfer task is broken down and represented as LAPP-RARs. The commands (RARs) marked with *Future capabilities* exceed the state-of-the-art capabilities of standardized lab robots. I do not elaborate on them in the present dissertation, but I cite relevant research and development endeavors, where applicable. During processing the results, I named each command by a *Command Identifier* and specified the corresponding hierarchical level.

2.4.3 Pick-and-place activities as LAPP RARs

In this section, I elaborate on a comprehensive representation of the pick & place labware transportation activities.

Definition of the RARs

First, I define the RARs in the following tables: T *Tasks* in Table 2.4, S *Subtasks* in Table 2.5, Q *Motion sequences* in Table 2.6, P *Motion primitives* in Table 2.7 and A *Actuator primitives* in Table 2.8. The parameters of each command are set in *italic*.

TABLE 2.4
TASK-LEVEL ROBOTIC ACTIVITY REPRESENTATIONS FOR PICK-AND-PLACE LABWARE TRANSFER

Command ID	**Description**
LabwareTransfer	Robot moves an object (of a certain *labwareType* e.g., `DEEP96`) from a predefined *source* site (a.k.a. nest) to a desired *destination* site. Depending on the distance between the two sites, this can be completed by a fixed base, rail-mounted, or mobile manipulator.
TeachBaseWaypoint	Store the current map position as an *baseWaypoint* for the Autonomous Mobile Robot (AMR) base.
TeachArmWaypoint	Store the current arm position as an *armWaypoint*. In the case of fixed base robots, this is expressed with relation to the world coordinates, whereas for mobile manipulators fiducial markers are used as a device-attached reference.
TeachLabware	Store current finger position (width) or current finger force under a specific *labwareType*
MaintainBattery	Maintain the battery charge level above a set *minPercentage* by sending the robot to Charge, if the battery charge level drops below the threshold.

TABLE 2.5
SUBTASK-LEVEL ROBOTIC ACTIVITY REPRESENTATIONS FOR PICK-AND-PLACE LABWARE TRANSFER

Command ID	**Description**
PrepareForInput	Prepare for picking, e.g, by raising the manipulator arm to a stand-by position.
GetLabware	Pick up a piece of labware (of a specific *labwareType* from a designated *source* site).
InternalPlace	Place labware on the robot's built-in storage (hotel).
PrepareForOutput	Prepare for placing.
InternalPick	Pick labware from the robot's built-in storage (hotel).
PutLabware	Place a piece of labware (of a specific *labwareType* to a designated *destination* site).
Charge	Send the robot to its charging station, make it dock and start charging.

The decomposition and sequence I present in this chapter adapt the Standardisation in Laboratory Automation Consortium (SiLA) features [84] to mobile manipulator robots. See section 5 for more details on how the various aspects of the LAPP framework are incorporated into real-life implementations and manifested in standardization endeavors, such as the activities of the SiLA Robotics Working Group.

I do not discuss representations lower than actuator primitive (such as position and force control activities), because they are not relevant from the perspective of the laboratory automation workflow. Instead, they are implemented on the low-level (embedded) robot control components.

The composition of pick-and-place RARs

Next, in Figure 2.1, I visualize the decomposition of the labware transfer (T) task into (S) subtasks, and further down into low-level RARs. The breakdown considers a mobile manipulator robot, consisting of an AMR base, a robot arm, and a gripper.

In table 2.9 I present how the pick-and-place RARs fit within a laboratory *service* and *workflow*. By putting fixed base robot arms, mobile robots and conveyor belts beside each

TABLE 2.6
MOTION SEQUENCE-LEVEL ROBOTIC ACTIVITY REPRESENTATIONS FOR PICK-AND-PLACE LABWARE TRANSFER

Command ID	**Description**
MoveThroughSequence	Robot arm performs the complete approach motion sequence (by visiting the corresponding armWaypoints in sequence) to finish at a given target *armEndpoint* (typically a site or its safe approach position).
DriveThroughSequence	Mobile base (AMR) performs the complete approach motion sequence (by visiting the corresponding baseWaypoints in sequence) to finish at a given target *baseEndpoint* .
OpenGripper	Robot opens the parallel gripper to prepare for gripping or to release the grasped object.
Grip	Robot grips an object (of a specific *labwareType*) with a parallel gripper.
Dock	Dock the AMR base to a specific *station* (e.g., with a laser scanner target)
Undock	Undock from a *station*.

TABLE 2.7
MOTION PRIMITIVE-LEVEL ROBOTIC ACTIVITY REPRESENTATIONS FOR PICK-AND-PLACE LABWARE TRANSFER

Command ID	**Description**
MoveToArmWaypoint	Robot arm moves to a specified *armWaypoint* by the means of a joint-optimized *movement* (MoveJ) or via a linear path (MoveL).
DriveToBaseWaypoint	AMR drives to a specified intermediary map position (*baseWaypoint*).
SetFinger	Gripper closes to a specified *force* or *width*.

TABLE 2.8
ACTUATOR PRIMITIVE-LEVEL ROBOTIC ACTIVITY REPRESENTATIONS FOR PICK-AND-PLACE LABWARE TRANSFER

Command ID	**Description**
SetJointPosition	Set a specific *joint* of the kinematic chain to a specific *angle* or linear *position* (depending on the type of joint).
SetFingerJoint	Set a specific gripper *joint* to a specific *angle* or *position* (depending on the type of gripper).

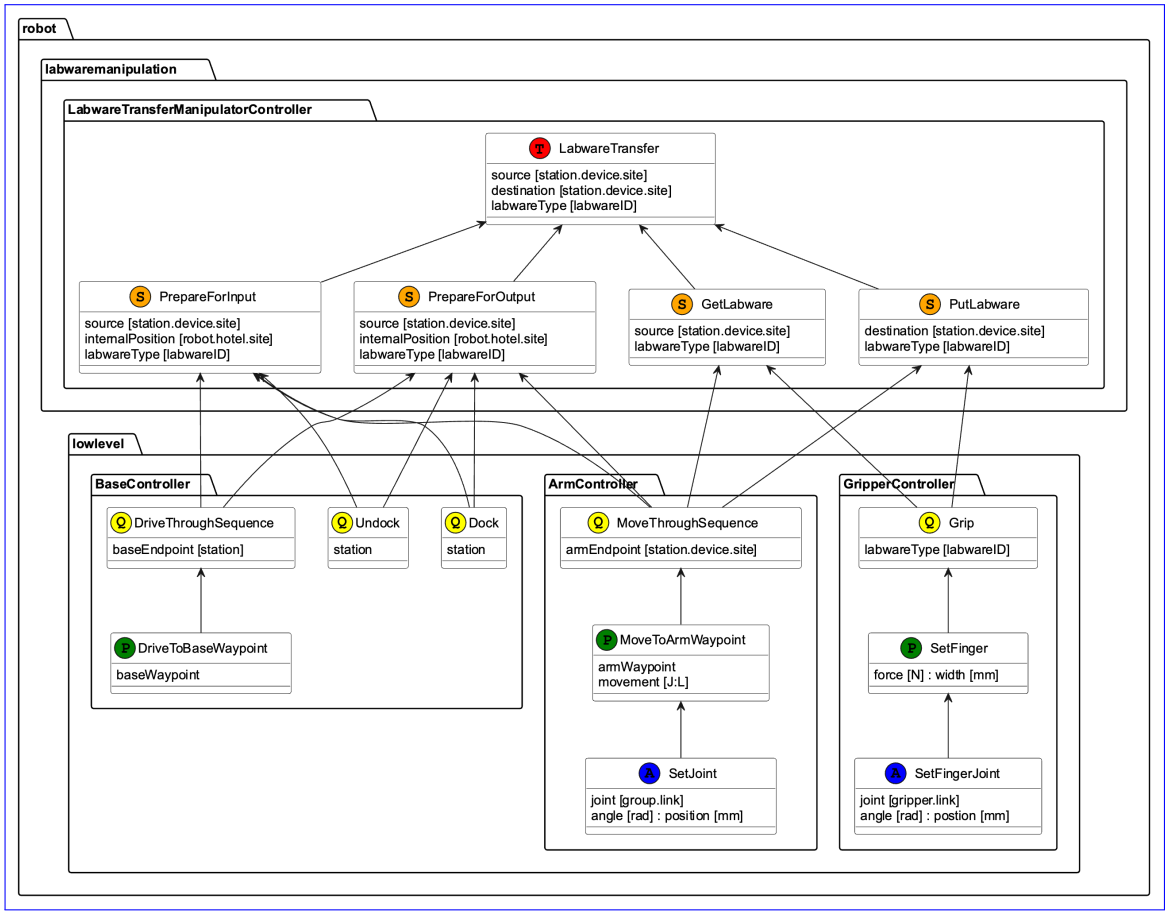


Fig. 2.1. UML-like representation of the `LabwareTransfer` RARs. Namespaces represent a way of categorizing the RARs. The arrows represent how a high-level RAR is composed of (broken down into) low-level RARs. Task (T), Subtask (S), Motion sequence (Q), Motion primitive (P) and Actuator primitive (A) are represented as classes. Parameters are shown as class members, with the data type in square brackets []. Overloaded parameters (in the case of polymorphic commands) are separated by a colon :

TABLE 2.9
BREAKDOWN FOR LIQUID HANDLING, ROBOT ARM, MOBILE ROBOT, AND CONVEYOR ACTIVITIES.

Lvl.	Liquid handler	Robot arm	Mobile robot	Conveyor
7	microscale services			
6	microscale chromatography workflow			
5	liquid transfer	labware transfer		
4	aspirate	get labware, put labware		
3	approach site	move to site	drive to station	-
2	motion vectors	move to arm waypoint	navigate to intermediary	move tray to desired position
1	pump control	joint control	base velocity commands	motor or magnet control

other, it can be seen that a labware transfer (T) task (and its corresponding (S) subtasks) can be implemented by different means. In addition, I show the analogy between the breakdown of a liquid handling activity and the pick-and-place RARs, building upon the considerations discussed in section 2.3.2.

The sequence of pick-and-place RARs

In the code snippet 2.1 and in Table 2.10, I map the pick-and-place RARs to the hierarchical levels presented in 2.3. The indentation in the code snippet 2.1 represents the decomposition levels defined in 2.3. I omit the *Service* and *Procedure* levels, as a `LabwareTransfer` T *Task* is non-specific to *Services* and *Procedures*; instead, it is a reusable unit of high-level lab robot activity.

The implementation considering a fixed base robot can be expressed by omitting the activities related to AMRs (marked with *), such as navigation. It can be seen that the activities of the robot arm and those of the AMR are analogous.

In a typical scenario, the scheduler would coordinate the *LabwareTransfer* T task by calling the *PrepareForInput*, *GetLabware*, *PrepareForOutput* and *PutLabware* S subtasks on the robot, in sequence. In the meantime, the source and destination devices should also be prepared for output or input, respectively. In the case of a mobile manipulator, the *PrepareForInput* S subtask includes undocking from the previous station (e.g., charger), driving to the next station via a sequence of waypoints, and docking there. After this, the arm needs to be unfolded into a safe/approach position. Note that this latter motion sequence is also performed by fixed base robots.

When the source device is ready for output, the *GetLabware* S subtask command makes the arm perform the final approach. This is either a fixed position, relative to the robot base (in case of fixed base robots) or a relative position (in the case of mobile manipulators), as described in [WA4]. The gripper then engages with the labware, and then the arm is retracted to a standby position ready for manipulation. This is followed by a Mobile Manipulator robot (MoMa)-specific internal S subtask to place the labware on the on-board storage (hotel).

When the *GetLabware* S subtask is completed, the source device is notified, and, with the *PrepareForOutput* command, the robot goes on to prepare for placement at the destination device. In case of a MoMa, this involves undocking from station 1, driving to station 2 via a sequence of waypoints, docking, and picking up the labware from the hotel. (Note the self-composition relation, similar to the placement on the hotel.) Then, the arm approaches a safe position to prepare for the actual placement upon command from the scheduler. This, again, is the same in the case of fixed base robots. Finally, the *PutLabware* S subtask is entirely the same for fixed base and mobile robots, and it involves the final approach, releasing the labware and retracting it to a standby position.

Source Code 2.1. Decomposition of a pick-and-place labware transfer task of a mobile manipulator robot

```
Legend
_____________________________________________________________________
Indentations represent the levels of the decomposition hierarchy:
Task
    Subtask
        MotionSequence
            MotionPrimitive
                ActuatorPrimitive
* Starred activities are specific for mobile manipulators
```

```
10  ____________________________________________________________________________
11
12  LabwareTransfer (station_1.device_1.site_1, station_2.device_1.site_1,
    ↪  DEEP96)
13      PrepareForInput (station_1.device_1.site_1, self.site_1, DEEP96)
14          *Undock (charger_1)
15          *DriveThroughSequence (station_1)
16              *DriveToBaseWaypoint (station_1.baseWaypoint_1)
17              (...)
18              *DriveToBaseWaypoint (station_1.baseWaypoint_n)
19              # Final baseWaypoint = station_1
20          *Dock (station_1)
21          MoveThroughSequence (station_1.device_1.site_1_safe)
22          # Safe position
23              MoveToArmWaypoint
                ↪  (station_1.device_1.site_1.armWaypoint_1, J)
24                  SetJoint (joint_1, <angle>)
25                  (...)
26                  SetJoint (joint_n, <angle>)
27              (...)
28              MoveToArmWaypoint
                ↪  (station_1.device_1.site_1.armWaypoint_n-1, J)
29                  # Safe position, aka. site_approach
30      GetLabware (station_1.device_1.site_1, DEEP96)
31          MoveThroughSequence (station_1.device_1.site_1)
32              # Handover position
33              MoveToArmWaypoint
                ↪  (station_1.device_1.site_1.armWaypoint_n, L)
34              # Final armWaypoint = site_1
35          Grip ()
36              SetFingers (<force>)
37          MoveThroughSequence (manipulation-ready)
38              MoveL (device_1.site-approach_1)
39              MoveL (device_1.device-approach)
40              MoveL (manipulation-ready)
41          *InternalPlace (self.hotel.site_1, DEEP96)
42      PrepareForOutput (station_2.device_1.site_1, self.site_1, DEEP96)
43          *Undock (station_1)
44          *DriveThroughSequence (station_2)
45              *DriveToBaseWaypoint (station_1.baseWaypoint_1)
46              (...)
47              *DriveToBaseWaypoint (station_1.baseWaypoint_n)
48              # Final baseWaypoint = station_2
49          *Dock (station_2)
50          *InternalPick (self.hotel.site_1, DEEP96)
```

```
51          MoveThroughSequence (station_2.device_1.site_1_safe)
52          # Safe position
53              MoveToArmWaypoint
                ↪  (station_2.device_1.site_1.armWaypoint_1, J)
54              (...)
55              MoveToArmWaypoint
                ↪  (station_2.device_1.site_1.armWaypoint_n-1, J)
56              # Safe position, aka. site_approach
57      PutLabware (station_2.device_1.site_1, DEEP96)
58          MoveThroughSequence (station_2.device_1.site_1)
59          # Handover position
60              MoveToArmWaypoint
                ↪  (station_2.device_1.site_1.armWaypoint_n, L)
61              # Final armWaypoint = site_1
62          OpenGripper ()
63          MoveThroughSequence (manipulation-ready)
64              MoveL (device_1.site-approach_1)
65              MoveL (device_1.device-approach)
66              MoveL (manipulation-ready)
```

2.4.4 Extended taxonomy

In table S1 of Appendix S2, I present the extended RAR taxonomy. Here, the scope is broadened beyond the simple pick-and-place labware transfer, and such capabilities are also included that are not yet fulfilled by state-of-the-art laboratory robots. Where appropriate, I cite relevant ongoing research and prototype products. The list is non-exhaustive and subject to future extensions and amendments, depending on the advancements in lab robot capabilities. This taxonomy was implemented as a skeleton structure for the SiLA feature framework [85].

RAR					Digital twin
Task	Subtask	Internal subtask	Motion sequence	Motion primitive	LAPP-DT position
LabwareTransfer	PrepareForInput	-	Undock	**DriveToBaseWaypoint**	PoI: Charger
				DriveToBaseWaypoint	PoI: Charger-approach
			DriveThroughSequence	**DriveToBaseWaypoint**	PoI: StationA-approach
			Dock	**DriveToBaseWaypoint**	PoI: StationA
			MoveThroughSequence	-	Pos: home (h)
				-	Pos: device-approach (s1d)
				-	HeadPos: down
			OpenGripper	**SetFingers**	GripperPos: open
	GetLabware		MoveThroughSequence	**MoveToArmWaypoint**	Pos: site-approach (s1s)
				MoveToArmWaypoint	Pos: hand-over pos (s1)
			Grip	**SetFingers**	GripperPos: close
			MoveThroughSequence	**MoveToArmWaypoint**	Pos: site-approach (s1s)
		Internal Place	**MoveThroughSequence**	-	Pos: hotel
			OpenGripper	**SetFingers**	GripperPos: open
			MoveThroughSequence	-	Pos: home (h)
	PrepareForOutput	-	Undock	**DriveToBaseWaypoint**	PoI: StationA-approach
			DriveThroughSequence	**DriveToBaseWaypoint**	PoI: StationB-approach
			MoveThroughSequence	-	Pos: home (h)
			Dock	**DriveToBaseWaypoint**	PoI: StationB
		Internal Pick	**MoveThroughSequence**	-	Pos: device-approach (s2d)
				-	HeadPos: down
				-	Pos: hotel
			Grip	-	GripperPos: close
			MoveThroughSequence	-	Pos: device-approach (s2d)
	PutLabware	-	MoveThroughSequence	**MoveToArmWaypoint**	Pos: site-approach (s2s)
				MoveToArmWaypoint	Pos: hand-over pos (s2)
			OpenGripper	**SetFingers**	GripperPos: open
			MoveThroughSequence	**MoveToArmWaypoint**	Pos: site-approach (s2s)
			-	-	Pos: home (h)

TABLE 2.10

DECOMPOSITION OF THE LABWARE TRANSFER SEQUENCE. HERE, THE DT POSITIONS ARE DISPLAYED ONLY FOR REFERENCE. THEY WILL BE DEFINED IN CHAPTER 3.

2.5 New scientific results

Thesis 1

I created a semantic decomposition scheme of workflows in life science laboratories, based on assessing the manual and semi-automated tasks in pharmaceutical process development laboratories.

Sub-thesis 1.1

I defined a general framework for the hierarchical decomposition of laboratory workflows, considering their automation and robotization. The framework characterizes the layers of the decomposition using the basic principles of Mutually Exclusive and Collectively Exhaustive (MECE). It draws inspiration from other domains and comprehends state-of-the-art laboratory automation capabilities.

I define the general layers of the hierarchical decomposition as follows:

7) **Service** refers to the entirety of the laboratory's capabilities, e.g., high throughput and/or microscale services.

6) **Procedure** is an experiment, an assay, or a repetitive/continuous laboratory process.

5) T **Task** is an elemental action item that is carried out by a human or a certain device.

4) S **Subtask** is an intermediary layer that represents parts of a task, that accomplishes minor landmarks.

3) Q **Motion sequence** is defined for the specific case of robotics, whereby the robot performs a sequence of motions, for example, to approach a handover site.

2) P **Motion primitive** is an elemental motion of a robot or other mechanism.

1) A **Actuator primitive** is an output excerpted by a certain actuator, e.g., robot joint, pump, etc.

0) **Physical process** refers to the actual subject of the laboratory activity, e.g., a set of samples, buffers and/or reagents.

Sub-thesis 1.2

I created a comprehensive breakdown of the primary supportive laboratory robot activity of pick-and-place labware transportation, as seen in Table 2.11. I defined exhaustive Robotic Activity Representations (RARs) for each level and formulated the compositional and sequential relations between them. In addition, I constructed an extended taxonomy of laboratory robot activities, including activities, that are not yet covered by existing solutions.

Related publications: [WA1, WA3, WA5, WA6, WA7]

RAR				
Task	Subtask	Internal subtask	Motion sequence	Motion primitive
LabwareTransfer	PrepareForInput	-	Undock	**DriveToBaseWaypoint**
				DriveToBaseWaypoint
			DriveThroughSequence	**DriveToBaseWaypoint**
			Dock	**DriveToBaseWaypoint**
			MoveThroughSequence	-
				-
				-
			OpenGripper	**SetFingers**
	GetLabware		MoveThroughSequence	**MoveToArmWaypoint**
				MoveToArmWaypoint
			Grip	**SetFingers**
			MoveThroughSequence	**MoveToArmWaypoint**
		Internal Place	**MoveThroughSequence**	-
			OpenGripper	**SetFingers**
			MoveThroughSequence	-
	PrepareForOutput	-	Undock	**DriveToBaseWaypoint**
			DriveThroughSequence	**DriveToBaseWaypoint**
			MoveThroughSequence	-
			Dock	**DriveToBaseWaypoint**
		Internal Pick	**MoveThroughSequence**	-
				-
				-
			Grip	-
			MoveThroughSequence	-
	PutLabware	-	MoveThroughSequence	**MoveToArmWaypoint**
				MoveToArmWaypoint
			OpenGripper	**SetFingers**
			MoveThroughSequence	**MoveToArmWaypoint**
			-	-

TABLE 2.11

DECOMPOSITION OF THE LABWARE TRANSFER SEQUENCE, THESIS 1.2

Part 3

DIGITAL TWIN FRAMEWORK FOR LABORATORY ROBOTIZATION

3.1 Motivation

Currently, supportive laboratory robots are primarily used for labware transportation, both in fixed base and mobile formats. These robots require a complicated setup procedure for each device or workstation with which they interact. Setting up the physical and logical interface between the robot and the device includes manually teaching the positions and movements for the Autonomous Mobile Robot (AMR) and the robot arm and setting up the control interface and the corresponding control sequence. The highly specialized technicians and scientists who work in life sciences laboratories are experts in fields such as biotechnology, chemistry, or material sciences. However, automation, and especially robotics, are fields where user experience needs to be streamlined. This enables laboratory experts to put their knowledge into the most important aspects of science and experimentation. Instead of having to deal with manual teaching and a complicated set-up procedure, a plug-and-play system is desired. In this way, the barrier for the implementation of laboratory automation can be reduced, which is a crucial aspect in terms of change management.

To achieve this, I propose an information model based on the Digital Twin (DT) approach and an operating model for the set-up procedure.

3.2 Literature and the state-of-the-art

3.2.1 Online teaching of laboratory robots

In the current state of technology, robotic manipulators in laboratory automation are mostly intended for transporting standard ANSI/SLAS-conform[1] microplates between different laboratory devices. To make this possible, in most cases, online teaching is performed when the robotic system is set up. This means that the positions are manually set by moving the robot by hand or jogging it with the controller.

In the case of mobile manipulators, the positions are defined in relation to Fiducial Markers (FMs), which is in most cases an optical Augmented Reality (AR) marker being

[1] Meets the Standards ANSI/SLAS 1-2004 through ANSI/SLAS 4-2004

detected by an RGB camera[47]. In certain cases, however, alternative perception sensors may be employed that can provide sufficiently accurate six-degree-of-freedom localization of either the marker or the handover position directly. These include RGB cameras extended with depth perception (e.g., time-of-flight sensors), LiDAR systems, reflective markers, or even the use of the robot arm itself with integrated proximity sensors to physically touch a fiducial marker.

The detection of the hand-over position relative to the robot's base is necessary because the precision of the mobile base is insufficient for the positions to be defined in a world-fixed Coordinate System (CS).

The robot keeps track of its coordinate systems (frames) by maintaining a transfer tree, where the relations between the frames are expressed. For example, the pose of the camera frame and the Tool Center Point (TCP) are both known in relation to the robot's base frame. When the arm moves, the system continuously updates the corresponding chain of frames based on the kinematic model and the joint angles (forward kinematics).

The DT approach

The present-day laboratory robotic setups pose a significant limitation regarding the manual setup procedure. To overcome this, an asset-centric information representation framework has to be introduced. The approach of a virtual representation of physical entities aligns with the DT concept. This section provides an overview of the DT concept, and the relevant nomenclature is specified.

Originally, this concept was proposed by Michael Grieves at NASA [86] for product life cycle management purposes. Maintaining a virtual representation of a *physical entity* and implementing bidirectional data connections enables various virtual operations ranging from modeling through testing to optimization. All these operations correspond to a specific sub-space of the DT. Generally, data are fed from the real space to the virtual space, and information is fed back alongside processes. A DT accompanies the product throughout its entire life cycle and can take different forms according to the specific stage. In parallel to the design phase of the product, a *prototype* of the DT is also developed, which is then *instantiated* for each physical product individually. The whole set of these instances comprise the *aggregate*, and they all reside in the corresponding *environment*. The *state* of both the physical and the virtual entities are stored in the form of *parameters*. These can either be static (so-called *prototype*) parameters which have the same value for each piece of the same product, or they can correspond to a specific physical specimen, in which case they are called *instance* parameters.

Since the proposal of the original concept, the DT notion has been adapted to a wide variety of use cases. Jones et al. [5] provide a systematic review of the related literature by thematic analysis and characterization. The paper also consolidates the terminology that I will follow in the context of the present thesis. Jones et al. also specify the parameters often used for DTs, summarized in Table 3.1.

3.2.2 Offline teaching with the DT approach

The DT concept is broadly applied for smart manufacturing use cases [87, 88, 89, 90], and there are also numerous examples of robotics applications, where the DT may have dif-

TABLE 3.1
PARAMETERS OF THE DT [5].

Parameter	Characteristics
Form	The geometric structure of the entity Can contain dimensions, CS definitions and 3D models
Functionality	The entity's capabilities, movement or purpose, Definition of the control interface and parameters
Health	State of the entity with respect to its ideal state
Location	Position and orientation (pose) With respect to the environment, layout or other entity
Process	Activities in which the entity is engaged, Scheduling parameters and models
Time	Duration to complete an activity, Timestamp
State	Measured (live) values of entity and environment parameters
Performance	Measured operation with respect to the optimal operation
Environment	The physical and virtual environment as a parameter
Miscellaneous	Requirements Qualification

ferent goals. These range from the education of future engineers [91, 92] through human-machine interactions [93, 94, 95] and control [96] all the way to programming [97, 25]. Tipary and Erdős propose a design approach that utilizes parametric DTs of robotic workcells for planning and programming [98, 99]. To adapt the offline-created robot program to the physical workcell, calibration and manual adjustments are needed in on-line mode most of the time. To bridge this gap, they define the so-called Digital Twin closeness as measuring the "geometric difference between the digital and physical counterparts of DTs for robotic workcells". Their DT comprises the models in Table 3.2.

TABLE 3.2
MODELS IN TIPARY'S ROBOT CELL DT [6]

Model	Content
Kinematic model	Geometry, kinematic behavior and component relations
Grasp model	Workpiece — manipulator relation through the gripper
Path model	Manipulation sequence and manipulator motion
Servo model	Condition-based manipulation (observation-based)
Metrology model	Providing information to other models and planner tools (observation-based)
Tolerance model	Represent the tolerance stack-up of the operation to assess its feasibility

3.3 The Laboratory Automation Plug and Play (LAPP) DT information model

I adopt the principles of DT-based offline teaching to the specific case of supportive laboratory robotics by defining an information model. As the DT concept has widely been adopted to smart manufacturing use cases, let us briefly review the analogies between

production and laboratory systems. I will provide an in-depth discussion of the system architecture aspects in chapter 4.

In a production system, the number one asset of interest, and the subject of the manufacturing process is the *product*. In contrast, in a laboratory setup, the subject of the experiment is the *sample*. Various pieces of equipment (either stand-alone or organized into *stations*) process the primary asset (product or sample) in both cases. Depending on the type of product, equipment can greatly vary in size and degree of complexity. In comparison, in a laboratory setup, especially in the context of plate-based assays, the format has lower variance. Laboratory devices include primary analytical instruments, means of processing, and supportive storage and transportation solutions. Supportive robots are intended to implement the material flow between these stand-alone units by means of the transportation of labware from the source device to the destination device.

3.3.1 General terms

I define the LAPP DT information model to facilitate the integration of laboratory equipment. I do this in an asset-centric fashion, considering the *device*, *room*, and *robot* categories. These three categories of assets make up a DT environment, where each asset instance is related to each other in terms of topology, position and function. I adapt the terms *DT prototype* and *DT instance* as introduced in section 3.2.1.

Parameters that belong to the *functionality* category include:

- The semantic description of the asset's capabilities may include Robotic Activity Representations (RARs) in the form of a control protocol's Application Programming Interface (API) definition (e.g., Standardisation in Laboratory Automation Consortium (SiLA) feature definitions or Laboratory and Analytical Device Standard (LADS) models).
- The specification of labware that the device can process or handle.

I consider the following *Form* parameters:

- Position definitions within the bounds of the device. I define these in detail in section 3.3.2.
- Collision geometry of the device, to enable dynamic robot motion planning.

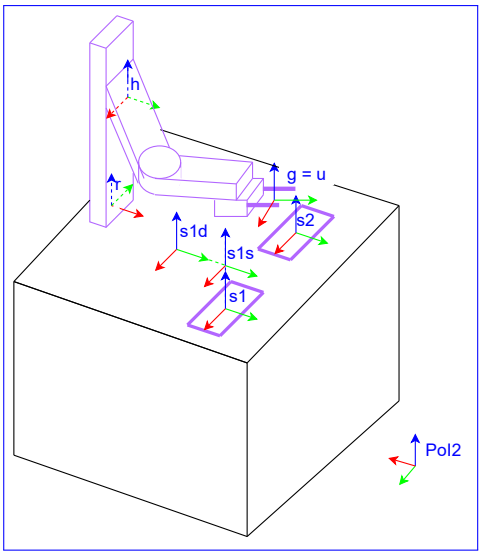
Finally, as a *miscellaneous* parameter, I consider the positional precision requirements specific to the device and specific robotic actions. Besides the categorisation above, I distinguish DT parameters in the LAPP framework as follows:

- **Prototype parameters** are unified for all specimens of the same make and model.
- **Instance parameters** override and extend the prototype parameters with instance-specific data.
  - **Static parameters** remain unchanged during the entire lifecycle of the instance. E.g., serial number.
  - **Volatile parameters** represent the current state of the instance.
    - **Observable parameters** represent real-time or semi real-time data streams. E.g., sensor streams, continuous progress updates.
    - **Unobservable parameters** are updated only upon specific request. For example, calibrated positions.

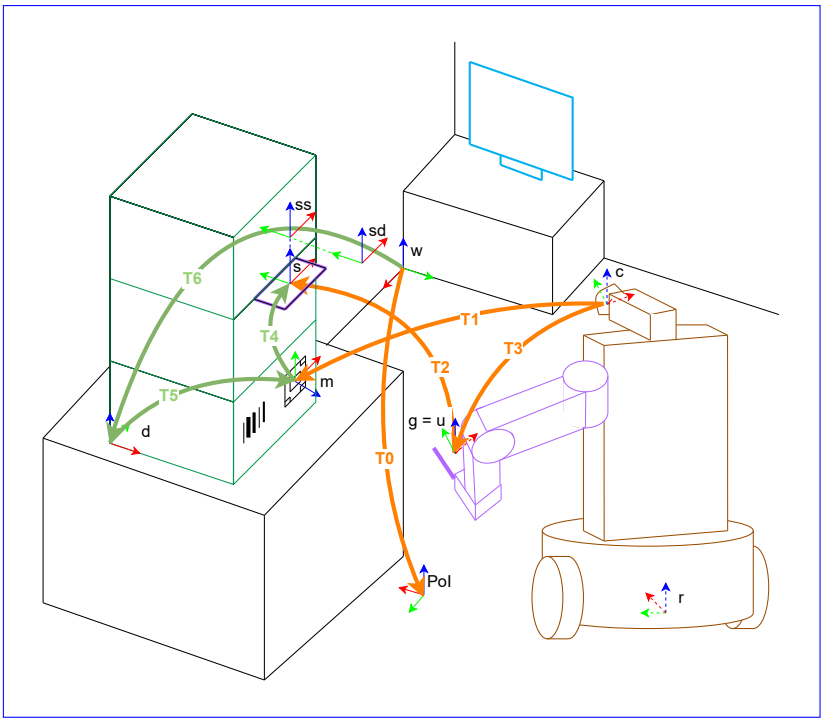
3.3.2 Geometrical parameters

In particular, I define the *form* and *location* type DT parameters as follows.

According to the LAPP approach, the positions (s, ss, sd in Fig. 3.1b) must be defined in relation to the marker frame (m in Fig. 3.1b) in cartesian space. These coordinate systems (frames) and position definitions fulfill the needs of typical laboratory robotics scenarios.



(a) Benchtop robot



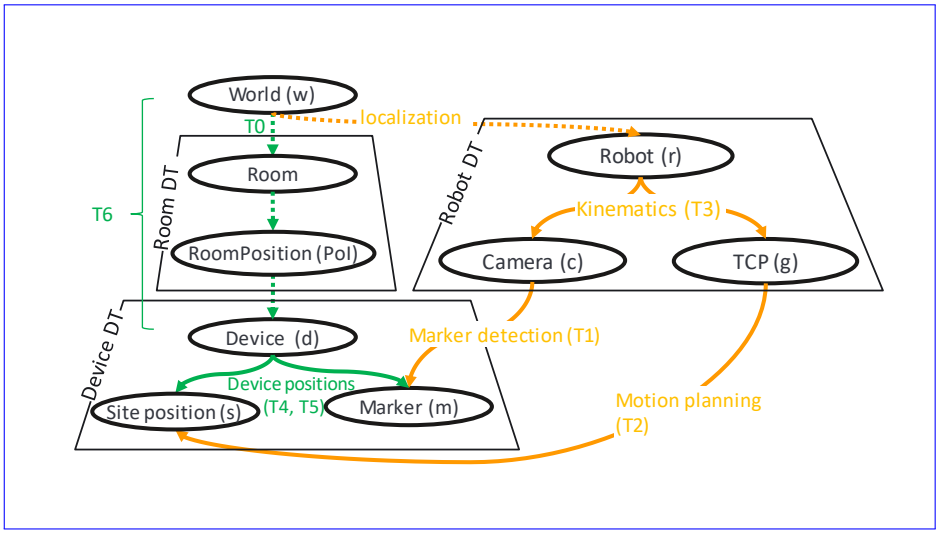
(b) Mobile robot

Fig. 3.1. Coordinate systems (i.e. frames of references) in the LAPP-DT environment. Coordinate frames are indicated with red, green, and blue axis triplet with their names indicated in black script. Arrows that connect these represent transformations, i.e., the expression of one frame relative to another.
Frames: d – device CS, g – robot's tool center point (TCP), h – robot home position, m – fiducial marker of the device, r – robot base, Point of Interest (PoI) – base pose for station, s – device handover site (plate nest), sd – device approach position, ss – site approach position.
Transformations: T0 - world-to-PoI, T1 camera-to-marker, T2 - TCP-to-site, T3 - camera-to-TCP, T4 - marker-to-site, T5 - device-to-marker, T6 - world-to-device), u – robot stand-by position, w – world.
Transformation color coding and line types: Orange – live robot-level transformations; Green – transformations stored in the LAPP-DT. Dashed – inaccurate transformations (from base odometry); Solid – accurate transformations (from robot kinematics, marker detection, or DT-stored positions).

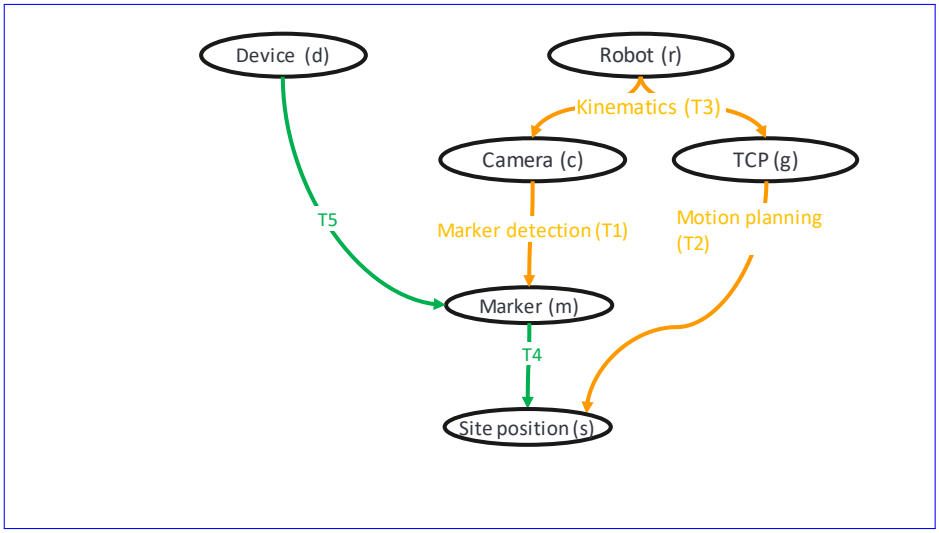
In the LAPP framework the positions are defined in relation to the marker frame in Cartesian space and not in the robot's CS in joint space. This enables a robot-independent implementation of the movements, thanks to the fact that the positions belong to a certain site of a workstation or device and not to one specific robot. This makes it possible to share positions between multiple robots. Ultimately, it can be assumed that a certain model of a laboratory device always has the same geometry, i.e., the handover site is at a pre-defined location of the device. Supposing the marker is also at a fixed and known position, the transformation T5 (as shown in Fig. 3.1b) can be deduced. The LAPP framework therefore requires the device vendor to include the marker in the device design. The marker must be visible on the same side from which the robot must access it. To ensure compatibility, a specific marker type and size must be specified. The currently available laboratory Mobile Manipulator robots (MoMas) [28, 29, 58] use different types of ArUco [47], AprilTag [100] and PI-Tag [101] markers of different sizes. Assessing these options and selecting a candidate is subject to future work.

The relations between the frames and positions in such a system depend on each other in that one frame is defined as a transformation of another. This means the frames can be represented in a graph structure, such as in Fig. 3.2, where the nodes are the frames and the edges are the transformations. The fact that fairly long chains of interlinked frames can form means that the uncertainties of each transformation can accumulate. These uncertainties are inevitably present both in the form of mechanical imperfections that cause deviations from ideal geometries and in the inherent inaccuracies of the metrology systems [102]. These must be considered by storing the uncertainty for each transformation in the DT. Also, a requirement in regard to precision must be specified for the specific robot action. This enables the process controller to match the requirement with the actual precision and determine if the specific system can perform the requested action.

It is also important to mention that the parent-child relations in a transform tree can change if a more accurate definition is available. For example, the following case can be considered: When the robot moves to a certain Point of Interest (PoI), the system can assume the approximate positions of the devices as defined in relation to the map $T_6(w \rightarrow d)$, as shown in Fig. 3.2a. However, the vision system can provide a far more accurate relation between the robot and the marker ($T_1(c \rightarrow m)$), which, along with $T_4(m \rightarrow s)$ can be used for the motion planning of the pick & place subtask. For this, a subtree from the robot frame (r) can be temporarily created, as presented in Fig. 3.2b.



(a) Global



(b) Robot-local

Fig. 3.2. Coordinate frame transformations (Markings of Fig. 3.1 are used.)

3.4 DT centric robot setup and operation with LAPP

Based on the asset-centric information representation framework proposed in section 3.3 I outline an operating model for the setup and usage of supportive laboratory robots.

3.4.1 Standard pick-and-place procedure

To be able to discuss the operation of labware transportation robots, I first derive a common Q motion sequence, as a common denominator.

As the simplest case, let us consider the operation of stationary robots. In this case, either the robot configuration is stored for each position in the form of joint values, or the position is directly prescribed in relation to the robot's base CS (r in Fig. 3.1a). Between these taught positions, different types of movements are possible. Usually, a sequence of

intermediary approach positions is defined for the robot before it reaches a final handover site with its gripper TCP. This ensures that no collisions occur between the robot and its surroundings, including the device it is supposed to load with samples. A typical sequence of steps for a pick-and-place task from the pick-up location site 1 (s1) to the drop-off location site 2 (s2) is presented below, using the coordinate systems in Fig. 3.1a:

- The robot starts in its home position (h).
- Performs a MoveJ-type movement to "untangle" itself and arrives at the standby configuration at the position (u).
- Moves linearly (MoveL) to a device-approach position (s1d).
- MoveL to a site-approach position (s1s).
- MoveL to the final site handover position (s1).
- Grips the plate.
- MoveL back to s1s.
- MoveL back to s1d.
- MoveL to s2d (not displayed).
- MoveL to s2s (not displayed).
- MoveL to s2 (not displayed).
- Releases the plate.
- MoveL back to s2s.
- MoveL back to s2d.
- MoveL to u and returns to standby/ready state.

I provided an in-detail definition of the pick-and-place labware transfer RAR for MoMas in section 2.4.3.

3.4.2 Harmonized operating model for manual teaching

The teaching process for marker-guided MoMas consists of the following steps, as derived from the feature definitions for Astech Projects' MoMa [103] and from the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA)'s Kevin Webinar [104]. The transformations in parentheses refer to the LAPP DT information model, as presented in section 3.3:

- The operator drives the robot to the station and makes sure the marker is in the camera's field of view.
- This pose of the mobile robot is stored as a PoI with the station's name $T_0(w \rightarrow PoI)$.
- The vision system determines the transformation from the camera frame to the marker frame: $T_1(c \rightarrow m)$.
- The operator moves the arm to the handover position so that the TCP and the site coordinate systems align: $T_2(g = s)$.

- Based on the robot kinematics model, the relation between the camera frame and the TCP ($T_3(c \rightarrow g)$) is determined.
- The relation between the marker and the handover site is calculated: $T_4(m \rightarrow s)$.

In a traditional setup, the resulting positions (coordinate transformations) would be stored in the context of one robot, expressed in joint coordinates. However, to enable the re-usability of the geometrical information, the LAPP DT approach should be used. Accordingly, the transformations shall be represented in a relational fashion, in the context of the *device*.

However, to minimize the burden on the system integrator and user, the LAPP proposal specifies the role of the device vendor as follows. It shall be the device vendor's responsibility to provide information shown in Table 3.3 as part of the DT prototype of the device. However, only in an ideal world would all vendors commit to making their devices LAPP-compliant. Especially for the launching and ramp-up period, the database will rely more on crowd-sourcing. For this, the DT prototype taxonomy must be provided as an open repository, with clearly defined contribution and admission mechanisms. Also, individual deviations in the device geometry may occur, thus the possibility for adjusting the DT instance by the means of calibration must also be kept open. This would involve a similar sequence as for manual teaching. See Section 3.5.

TABLE 3.3
PARAMETERS OF THE DEVICE'S DIGITAL TWIN PROTOTYPE. CS MARKINGS DEFINED IN 3.1

Piece of information	**As DT parameter (See Table 3.1)**
Robotic tasks, subtasks and motion sequences: e.g., transport by pick & place	Functionality
Type of workpiece: ANSI/SLAS microplate, Falcon tube etc.	Functionality
Location of the marker (m) in relation to the device CS (d)	Form
Simplified collision geometry of the device, defined in d	Form
Positions. e.g., handover site s, Site approach ss, Device approach sd	Form
Precision requirements	Miscellaneous

3.4.3 The LAPP autonomous setup sequence

With the help of the LAPP DT, a compatible MoMa with a calibrated camera system will be capable of performing the setup fully autonomously, as visualized in 3.3. After that, in ideal circumstances (i.e., with calibrated geometries), the pre-defined robotic actions can be performed immediately, without requiring manual input from the integrator or operator. This is the functionality referred to as plug & play.

As part of the sequence presented in [WA3], the coordinate transformations are determined and the DT instances are parameterized, as presented in Table 3.4.

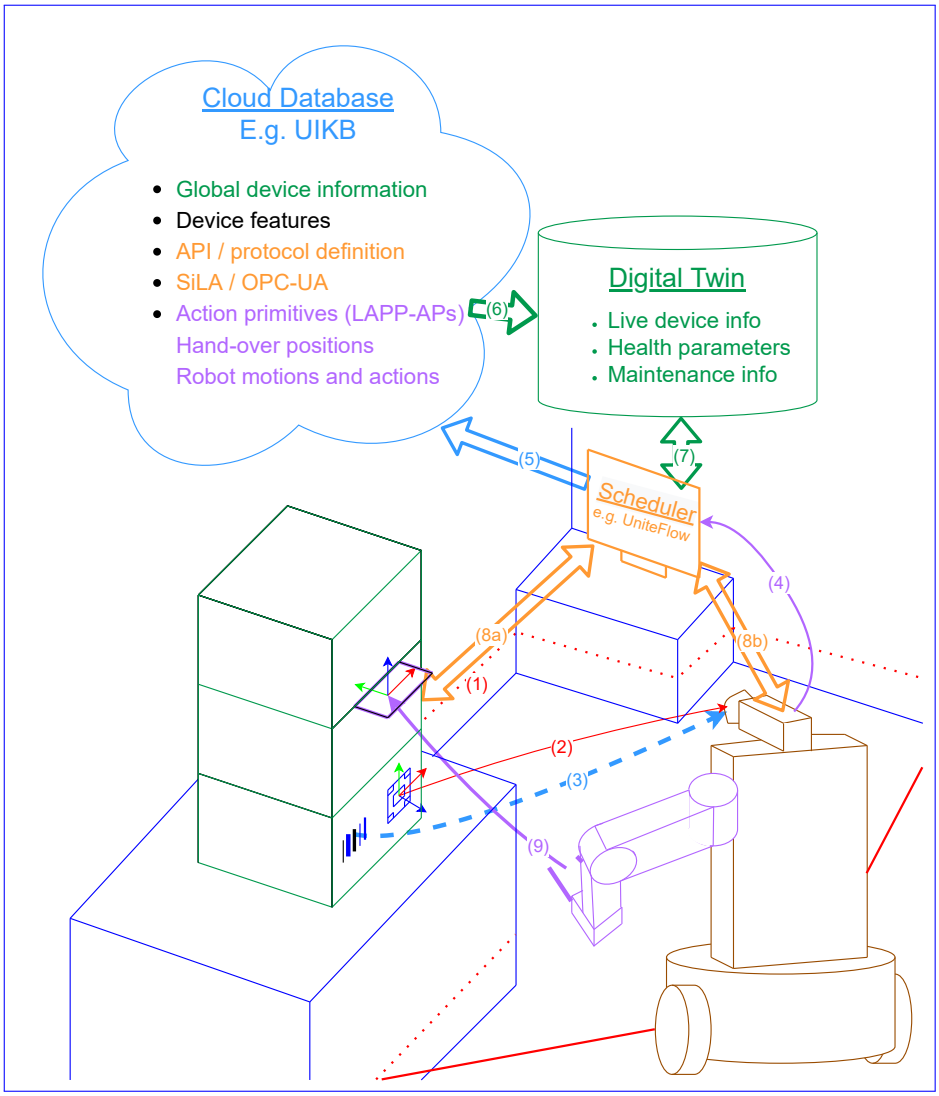


Fig. 3.3. The LAPP sequence (1) Generate map, (2) Detect barcode, (3) Detect fiducial pose, (4) Upload barcode ID to the scheduler, (5) Request device information from the cloud database, (6) Instanciate digital twin from the cloud database, (7) Keep digital twin updated, (8a) Device control, (8b) Robot control, (9) Robotic manipulation

TABLE 3.4
THE PLUG & PLAY SETUP SEQUENCE, MARKINGS OF FIG. 3.1B USED

Description	**Transform.**	**DT parameter type**	**DT instance**
During the autonomous room discovery procedure, the map is generated		Form	Room (instance)
Simultaneously, the approximate device positions are detected with the markers. Since d and m are already connected by the DT prototype of the device, and the robot is at the PoI, d is now defined in w.	$T_6(w \to d)$	Location	Device (instance)
The handover site position is taken from the DT prototype of the device	$T_4(m \to s)$	Form	Device (prototype)
If necessary, this position can be overridden (calibrated) and stored in the DT instance of the device.	$T_4, cal(m \to s)$	Form	Device (instance)
The robot kinematics can be recalibrated and stored in the DT instance of the robot	$r \to g$	Form	Robot (instance)

3.5 Challenges and key enablers

Traditional laboratory robots are usually taught in joint space, which means that the robot configuration is stored directly in the form of joint values for each position. This results in the fact that the accuracy of the robot's kinematic model is not crucial. On the contrary, when the positions are specified in world coordinates, the robot controller must perform inverse kinematics calculations to determine the corresponding joint values. For this, the robot's geometry must be precisely modeled, e.g., by Denavit–Hartenberg parameters or by the Unified Robot Description Format (URDF). These are specified initially for each robot model during the design process. Following the DT notation, these data correspond to the DT prototype of the robot model. Due to manufacturing imperfections, however, each piece of a finished robot has slightly different geometries in reality, which must be readjusted by means of calibration. These new parameters are stored in the DT instance for the certain robot. To achieve the sub-millimeter precision required for efficient and reliable plate manipulation, the possibility must be kept open for the robot to be re-calibrated.

An important distinction is between online vs. offline teaching. Online teaching means that the positions are taught directly on the physical robot by manually moving its end-effector to the desired positions. In this case, the robot's repeatability is more important than its absolute accuracy; what matters is that it is consistent even if systematic error exists. Offline teaching means that the positions are defined without the physical robot, e.g., in the case of the LAPP-DT approach by the device vendor. In this case, the absolute accuracy is just as important as the repeatability, since consistency is insufficient if the position is off. Therefore, the precision requirements towards a LAPP-enabled robot must be formulated adequately. For DT-based offline teaching scenarios in particular, the DT closeness to measure such precision is defined by Tipary et al. [99] (see Section 3.2.1).

Besides kinematics, the robot's mechanics also play an important role in the robot's precision. One such effect is that no mechanical system is perfectly rigid but flexible and compliant to a certain degree. Reasons behind this are, on one hand, the mechanical properties of materials: depending on a component's shape and its elastic modulus, it bends under load. On the other hand, the joints of robots consist of bearings and gears, which might have mechanical play or backlash. These two effects result in the whole structure diverging from the ideal position when under the effect of internal and external forces, which could be calculated by pure kinematics. These effects must be mechanically minimized and/or countered by a suitable controller.

Calibration might also be necessary with regard to the devices' geometries and the positions of the handover sites. As described in Section 3.4.3, this transformation (T_4) can be fetched from the DT prototype, but overridden (calibrated) on-demand in the DT instance.

3.6 New scientific results

Thesis 2

I propose an information model and an operating model for facilitating the integration of laboratory assets with automation systems and robots.

Sub-thesis 2.1

Based on the Digital Twin (DT) concept, I created an information framework to represent the parameters of laboratory assets, allowing their integration into an automated robotized ecosystem.

I adopted the canonical DT nomenclature and categorized the specific *functional*, *form*, and *miscellaneous* parameters for laboratory assets. Furthermore, I redefined the distinction between prototype and instance parameters.

Parameters that belong to the *functionality* category include:

- The semantic description of the asset's capabilities may include Robotic Activity Representations (RARs) in the form of a control protocol's Application Programming Interface (API) definition (e.g., Standardisation in Laboratory Automation Consortium (SiLA) feature definitions or Laboratory and Analytical Device Standard (LADS) models).
- The specification of labware that the device can process or handle.

I consider the following *Form* parameters:

- Position definitions within the bounds of the device. I define these in detail in section 3.3.2.
- Collision geometry of the device, to enable dynamic robot motion planning.

Finally, as a *miscellaneous* parameter, I consider the positional precision requirements specific to the device and specific robotic actions.

I created a specific information model for hierarchically representing geometric information to enable offline teaching of laboratory robots, as shown in figure 3.4.

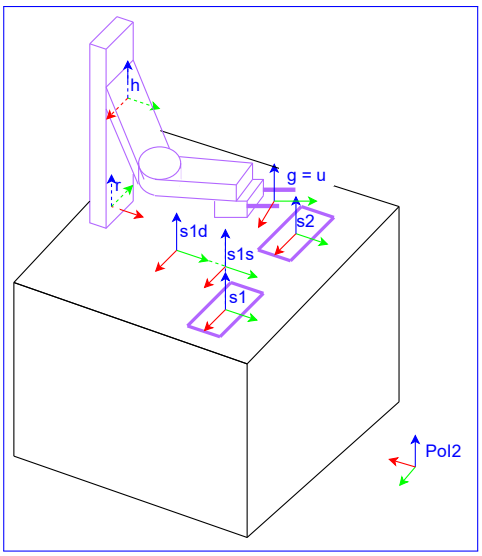
Sub-thesis 2.2

I derived a harmonized operation model for the DT-based setup and operation of supportive laboratory robots. An autonomous offline teaching procedure is enabled by utilizing the information models of sub-thesis 2.1 (see table 3.5).

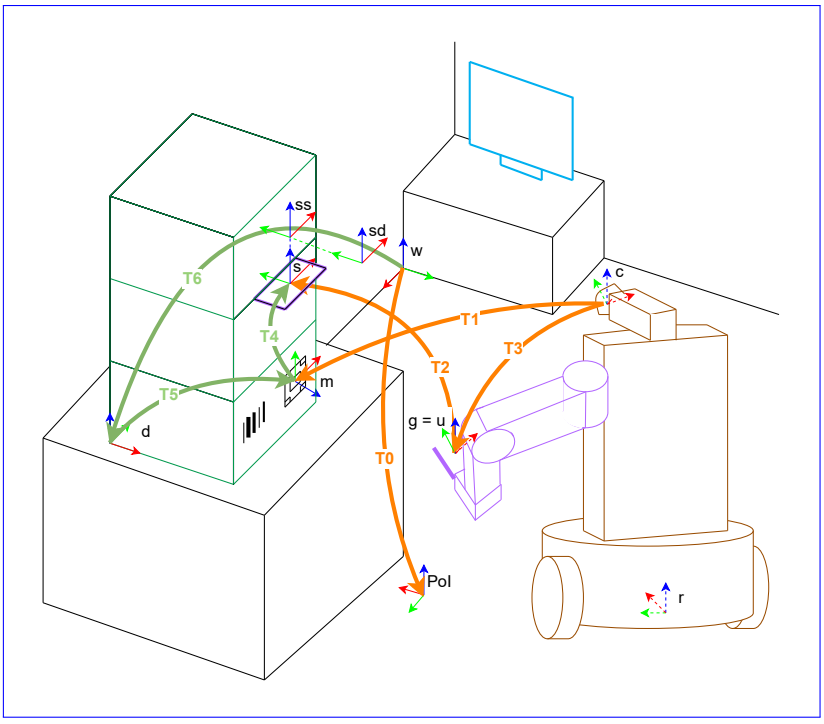
Related publications: [WA3, WA4]

TABLE 3.5
THE PLUG & PLAY SETUP SEQUENCE, MARKINGS OF FIG. 3.1B USED

Description	**Transform.**	**DT parameter type**	**DT instance**
During the autonomous room discovery procedure, the map is generated	w	Form	Room (instance)
Simultaneously, the approximate device positions are detected with the markers. Since d and m are already connected by the DT prototype of the device, and the robot is at the Point of Interest (PoI) d is now defined in w.	$T_6(w \rightarrow d)$	Location	Device (instance)
The handover site position is taken from the DT prototype of the device	$T_4(m \rightarrow s)$	Form	Device (prototype)
If necessary, this position can be overridden (calibrated) and stored in the DT instance of the device.	$T_4, cal(m \rightarrow s)$	Form	Device (instance)
The robot kinematics can be recalibrated and stored in the DT instance of the robot	$r \rightarrow g$	Form	Robot (instance)



(a) Benchtop robot



(b) Mobile robot

Fig. 3.4. Coordinate systems in the Laboratory Automation Plug and Play (LAPP) environment. d: device CS, g: robot's tool center point (Tool Center Point (TCP)), h: home position for robot m: fiducial marker for device, r: robot's base, Point of Interest (PoI): base pose for station, s: handover site (plate nest) of device, sd: device approach position, ss: site approach position, T0...T6: relational frame definitions (T0: world-to-PoI, T1: camera-to-marker, T2: Tool Center Point (TCP)-to-site, T3: camera-to-Tool Center Point (TCP), T4: marker-to-site, T5: device-to-marker, T6: world-to-device), u: stand-by position for robot, w: world.
Orange arrows: live robot-level transformations, Green arrows: stored in the Laboratory Automation Plug and Play (LAPP)-DT, Dashed arrows: inaccurate transformation (originating from base odometry), Solid arrows: accurate transformation (originating from robot kinematics, marker detection or DT-stored positions)

Part 4

REFERENCE ARCHITECTURE MODEL FOR THE INTEGRATION OF LABORATORY ROBOTS

4.1 Motivation

Laboratory automation systems are complex and heterogeneous, with diverse instruments and integration approaches. Unlike industrial automation, this domain lacks a standardized Reference Architecture Model (RAM) that prescribes a canonical architecture and best practices.

A RAM is a conceptual framework defining key system components, their relationships, and integration principles. In this thesis, the Laboratory Automation Plug and Play (LAPP)-RAM is introduced as a layered, modular representation of laboratory automation, covering devices, control systems, data layers, and interfaces. It provides a common vocabulary and design guide for stakeholders.

Industrial frameworks such as International Society of Automation (ISA)-95 or Reference Architecture Model Industrie 4.0 (RAMI4.0) show how a RAM can align stakeholders, improve interoperability, and reduce integration costs. Similarly, the LAPP RAM aims to streamline deployment, simplify scaling, and enhance long-term flexibility in laboratory automation.

Motivation

The goal for introducing the RAM in this part of the dissertation is to provide a unifying conceptual framework that connects the previously introduced semantic and information model aspects of the LAPP framework. Such models aim to capture the system architecture and software topology in a structured way, linking laboratory automation devices, orchestration layers, and supportive robotics with their respective control components.

A *Reference Architecture Model* serves as a blueprint: it harmonises and generalises diverse practices, outlines the generic roles and responsibilities in a multi-layer control architecture, and provides a consistent representation of complex system and software topologies. This is particularly important in laboratory automation, where the current landscape

is fragmented. The lack of standardisation across instrumentation, vendor-specific implementations, and the increasing variety of supportive automation solutions hinder seamless end-to-end integration. By providing a common frame of reference, a RAM enables different stakeholders to converge on shared principles and facilitates interoperability across heterogeneous systems.

An illustrative example from the broader industrial context is the Reference Architecture Model Industrie 4.0 (RAMI4.0). This three-dimensional layered model provides a fundamental architecture for Industry 4.0 systems, supporting both technical standardisation and practical adoption. Similarly, the LAPP-RAM follows these principles: it offers a basis for discussion, creates a common understanding of standards and norms, and supports the representation of assets (i.e., the components of the automated ecosystem) at varying levels of granularity. Corresponding information models capture the relationships among these assets, thereby enabling the implementation of a comprehensive digital twin ecosystem.

The usefulness of a reference architecture model is twofold. On the one hand, it supports the generalisation of best practices into a structured, reusable framework. On the other hand, it enables transferability of technical solutions across domains, technology maturity levels, and application contexts. For instance, the LAPP-RAM provides a modular and agnostic approach that can guide the progression from academic prototypes to industrial pilots and finally to production-level implementations, all while preserving the same underlying principles and architecture. This ensures that solutions remain adaptable to specific domains, while still benefiting from the coherence of a common framework.

Concrete examples of this applicability are presented in Chapter 5, where the LAPP concept is applied to distinct use cases that differ in principles, approaches, and operational circumstances. These cases demonstrate how the framework, by maintaining its generalised structure, supports both domain-specific adaptation and overall transferability.

4.2 Background

This section provides an overview of the established and emerging technologies in other fields, such as the mechanical manufacturing, process- or automotive industries. I highlight analogies and synergies along with naming some areas from where ideas could be adapted or have already been adapted to the needs of laboratory automation. As such, I will discuss the classical vertical integration approach, reviewing the different layers of industrial automation systems. To some degree, this approach has already been adapted to laboratory automation. However, the need for an intelligent infrastructure in combination with connectivity to different assets can only be achieved by adopting new horizontal and distributed integration approaches, such as the Internet of Things (IoT) or the RAMI4.0.

4.2.1 Industry agnostic evolution of automation systems

To get an overview of some of the building blocks of modern digitalized (*smart*) industrial production systems, the different levels and layers of planning, control, supervision, and data collection must be considered. In this regard, many similarities and analogies can be drawn between industrial production and laboratories [WA2]. At the highest level,

Enterprise Resource Planning (ERP) has to take place, while underneath a Distributed Control System (DCS) or a Supervisory Control and Data Acquisition (SCADA) resides. These systems feature several layers of scheduling, control and supervision ranging all the way down to the individual machines as the elemental components of the system.

End-to-end digitalization and equipment integration are crucial aspects in industrial automation and in the process industry. To address these, multiple endeavors took place to provide standardized protocols. One of these, the Open Platform Communications - Unified Architecture (OPC-UA), was introduced in 2008 and has since become a worldwide standard [105]. OPC-UA implements client-server communication with industrial equipment and systems for control and data collection. As an open standard with an increasingly complex specification, it enables vendors and organizations to model their own control and data structures into an OPC-UA name-space. As such, upon the infrastructure, information models provide a specific layer for industry standards and vendor information. The OPC Foundation is closely working together with industrial societies, such as the International Society for Pharmaceutical Engineering (ISPE), the German Mechanical Engineering Industry Association (VDMA) and Zentralverband Elektrotechnik- und Elektronikindustrie (ZVEI). The latter is hosting a large number of working groups for defining Companion Specifications for various domains [106]. The ISPE Pharma 4.0 group has the focus to add industry and regulatory-specific elements that are required for Good Manufacturing Practice (GMP) manufacturing of pharmaceutical products.

To facilitate the cooperation and collaboration of technical objects by means of virtual representation and connectivity, the RAMI4.0 [107] was created by the Plattform Industry 4.0 network in Germany. For this purpose, a Digital Twin (DT) approach is described where the physical world is represented in the information world. The central notion is the *asset*, which is broadly defined as an object that has value for an organization. It can be a physical entity (product, machine, machine element, etc.) or a virtual/logical entity (software, idea, archive, etc.).

Five concepts are defined within the context of an asset:

- **Presentation within the system**: Unknown, anonymously known, individually known or administered asset.
- **State within its lifetime**: Type (DT prototype using the nomenclature of [5]) or instance (same as [5]).
- **Communication capability**: Not capable, passively capable, actively capable, or I4.0 compliant.
- **Representation by means of information**: entirety of characteristic data and properties.
- **Technical function**

The RAMI4.0 defines a structure to describe the main elements of an asset along three axes:

- Arthitecture (Layers): Information relevant to the role of the asset. Can be interpreted as elements of the DT.
- Life cycle & value stream: State at a particular time and location
- Hierarchy levels: Extension of the automation pyramid approach

The RAMI4.0 spans a three-dimensional space. This means that every element of the automation pyramid can be considered as an asset and accordingly have its corresponding architectural layers and life cycle. The RAMI4.0 framework served as primary inspiration while outlining the LAPP-RAM.

Based on the RAMI4.0 model and intelligence infrastructure, the I4.0 Asset Administration Shell (AAS) concept was developed. It provides a structured interface for connecting I4.0 to physical things by encapsulating all data and information about the assets (active and passive). According to AAS, each physical asset, such as robots, sensors, analytical devices, and even workpieces, have their own admin shell representation. Assets can be organized into multiple layers of thematic groups, which have their own corresponding admin shell [108].

In chapter 2 I showed, that hierarchical decomposition is often used in project management to plan the scope, resources, scheduling, and timelines. To take it a step further, let us consider how the planning aspect can be bridged towards execution in an appropriate system architecture. This can be observed in the case of a Business Process Model and Notation (BPMN) process [109], which can be directly run on a process engine. When it comes to executing the process on an automation system, the levels of the process representation are often reflected in the levels of the control system. The levels of complex automation architectures are primarily delimited based on physical and technological considerations. In general, the upper layers influence the workings of the lower ones by means of control and the assignment of activities, while information flows from the bottom layers towards the upper ones. I discuss this aspect in 4.2.2.

4.2.2 Strict multi-level structure vs. service-oriented architecture

Multi-level architectures are well-established and widely applied in conventional industrial automation scenarios. The ISA-95 standard [110] provides a hierarchical model and terminology for the integration of ERP, Manufacturing Execution Systems (MES), and SCADA systems. On the bottom, the physical process is considered, which produces the desired outcome (product). This process is bi-directionally interfaced with the *Field* level, which is the level of sensors and actuators. In addition to this, there is usually a *Direct control* layer containing the Input-Output (IO) modules and low-level logic implemented on a Programmable Logic Controller (PLC) or microcontroller. The *Process supervisory* layer aggregates the information coming from the *Direct control* nodes and provides the operator with a Human-Machine Interface (HMI). The *Process supervisory* layer, which is responsible for the monitoring and control of the production process. The *Production Supervision* (also called Manufacturing Operations Management) layer controls the workflow or recipe, maintains records, and optimizes the production process. The highest level in this distinction is that of the *Plant Management and Scheduling*, which establishes the plant schedule and controls factors such as material use, delivery, and shipping and inventory levels. I drew an analogy between such industrial automation systems and laboratory automation architectures in [WA2].

In the context of the Industry 4.0 approach, the need for increased flexibility of processes and individualized production created the need for a new approach [111]. The reasoning is twofold. For information to reach the higher levels, it must propagate through multiple intermediary layers, often via multiple different protocols. Moreover, to adapt the

system to a new process, the low-level control units (in most cases PLCs) must be manually reprogrammed, taking into account their interlinked codependencies. In contrast, Schnicke et al. [112] adopt a peer-to-peer Service-oriented Architecture (SoA), according to the core principles of Industry 4.0. Stateless services are implemented, which can be called with the appropriate parameters by another component or an orchestrator. For that, information must be represented in an appropriate DT framework. In this framework, the elemental component is the service provider, i.e., a functional unit, such as a device or machine. For each service provider, the offered capabilities are described in the form of a *ServiceTag*. Services can be organized into sequences, i.e., recipes that prescribe the steps for manufacturing a product. This makes it possible that, in case of a change in the process, only the high-level process description must be adapted, and the low-level implementation steps are reconfigured by organizing them into a new order and/or changing their parameters.

Besides focusing on overcoming the strict multi-level (or automation pyramid) structure, Schnicke et al. consider multiple different ways of introducing hierarchical relations. First, they distinguish between *transforming services* and *supportive services*. The former means actions that actually change the product in some way, while the latter means non-transforming actions, such as transportation from one station to another. A high-level (abstract) recipe only prescribes the transforming actions that the product must go through, all in a device-agnostic fashion and without supportive actions. It is the orchestrator's role to turn this into an executable plan by taking into account the plant topology and the service providers' capabilities. This also means that the supportive services are filled in wherever needed, e.g., for a transfer between two stations. The second example of introducing hierarchical relations is the statement that services can be nested into each other. By this, a multi-level structure can be implemented both in the process representation aspect and also by the means of control. However, this optionally introduced hierarchical structure is not enforced to follow the strict levels of conventional automation pyramids. Thirdly, the possibility of exposing service details on multiple levels is mentioned. The given example is a transport service separated into *prepare*, *perform* and *reset* steps. Table 4.1 maps Schnicke's framework to a hierarchical decomposition.

TABLE 4.1
HIERARCHICAL LEVELS IN SCHNICKE'S FRAMEWORK, MAPPED AGAINST THE LEVELS OF LAPP (DEFINED IN SECTION 2.4)

Level	Process	Protocol	Control architecture
8			
7	Order		
6	Product recipes	BPMN	Orchestrator & process engine
5	Service	AAS	Control component (PLC)
4	Steps		
3			
2			
1			
0			

A similar consideration was also one of the key principles for designing the Standardisation in Laboratory Automation Consortium (SiLA) 2 protocol [113]. SiLA enables service-oriented, peer-to-peer communication, which aligns with the principles of Industry 4.0. In this framework, service descriptions are called *feature definitions*. Although it is beyond the scope of the present article to discuss the SiLA 2 protocol comprehensively, the protocol will be a key building block in implementing LAPP. The feature definition framework and its design principles are also relevant on a conceptual level, thus, I will discuss certain aspects.

4.2.3 Specifics to laboratory automation

Principles of traditional multi-layer industrial automation architectures and modern service-oriented paradigms have been adapted to the needs of laboratory automation in different ways.

Knobbe et al. [81] present and evaluate the AI.Laboratory concept on the example of the pipetting skill, which is considered as a core task in laboratory automation. They implemented a platform consisting of two of their so-called Intelligent Robotic Lab Assistants, which are 7-axis robot arms mounted on 2-axis gantries for range extension. They employ a finger changer, which lets the robot use specific grippers for hand-held electrical pipettes. Unlike other robotic solutions, their pipetting skill extensively complies with good pipetting practices in picking up the tips with the appropriate force, keeping the immersion depth while aspiring aspiration, dispersing against the wall, and wiping the tip afterward. With this technique, they could achieve a better accuracy than human operators. The system incorporates extensive safety features and also a DT in the form of a virtual 3D representation of the system, to which the telemetry data is connected. As such, form and state-type DT parameters are covered.

Knobbe's system architecture employs Industry 4.0 principles. The setup is centered around the robotic system, and the corresponding *Lab Skill System*. The robotic system, the external sensors and corresponding software services make up the *General Purpose Layer*, while the skill-centric aspect is considered as the *Application Layer*. The DT resides in the *Cloud Services* system component.

Another specific example, targeting the chemistry experiment automation domain is the ARChemist [114] framework. In comparison to other laboratory robot systems, such

as the robot chemist [52], which conducted a photocatalysis experiment series and solubility measurements supported with robotics and computer vision [115], the ARChemist provides a platform not only for one single type of experiment, but a reconfigurable flexible architecture. This can integrate different kinds of robots and enables the reconfiguration of the experimental setup to adapt to the needs of the experiment. The system has an interface for the chemist, the main element of which is the chemistry recipe, which is a human-readable prescription of what needs to happen to the sample. The ARChemist system parses this along with the configuration file, which describes the experimental setup: the available stations and robots.

LAPP shows a considerable overlap with the ARChemist, in terms of the goal statement and the main aspects of the solution. In the ARChemist framework, each station and robot have their own operational descriptor, which essentially serves as a semantic service description. In comparison to that, the LAPP Robotic Activity Representations (RARs) adds the dimension of the hierarchical decomposition and an in-depth prescription of the labware transfer activity (chapter 2). The ARChemist system also has a state representation framework, which resembles the DT approach. It keeps track of the physical and operational status of resources. The LAPP DT concept details the information model and outlines an operation model for the integration and usage of laboratory robots. The ARChemist framework outlines a system architecture, based on the Robot Operating System (ROS) middleware. The process starts with the chemist defining the configuration and the recipe, which get parsed by the *Persistance Manager* into the *State Manager*. Based on the recipe, the *Workflow Manager* conducts the experiment, and assigns the samples to stations. The *Robot Scheduler* submodule interfaces with the ROS *Robot handlers*, while the *Workflow Processor* with the *Station handlers*. The system is laid out in a service oriented fashion, and does not feature a strict hierarchical structure. The LAPP RAM draws inspiration from traditional, hierarchical industrial automation principles, and flexible Industry 4.0 approaches.

4.2.4 Process control and experiment orchestration

When discussing the laboratory automation domain, it is important to distinguish between two major sub-domains: process control and workflow orchestration. As for the aspect of the system architecture, let us consider their specifics and their overlaps.

Advanced Process Control (APC)

As discussed in chapter 1, APC refers to contained, continuous or batch-operated systems, where material is primarily transmitted by the means of pipes and tubes. In these systems, the focus is on time-series data, as in SCADA systems. However, on top of that, advanced Process Analytical Technologies (PATs) are used to implement the control loop feedback, which include non-real-time techniques, such as High-performance Liquid Chromatography (HPLC) or mass spectroscopy. These techniques necessitate specialized APC tools, such as PharmaMV [116]. These software tools provide multivariate analysis and real-time pharmaceutical process control capabilities. They adopt many approaches and techniques from industrial automation (e.g., semiconductor manufacturing and process industries),

as such, the principles of a SCADA systems. Also, established industrial interoperability protocols, such as OPC-UA are widely used by pharmaceutical APC tools.

In addition to the previously mentioned non-real-time PATs, some parameters can only be measured in multi-step at-line or offline fashion. This means that the samples need to undergo an experiment consisting of several steps. This setup already represents the category of workflow orchestration.

Workflow orchestration

In contrast to contained (pipe-based) real-time or semi-real time APC systems, in the case of orchestrated workflows, the transport of material (including samples) takes place in various containers (labware). These containers need to be passed from station to station, from device to device, for the steps of the assay to be completed in sequence. The over-arching orchestrator layer, in this case, is responsible for interpreting the experiment description ("recipe"), and breaking it down to the parts of the system. The scheduler component assigns the samples and materials to the respective devices, and coordinates their flow between the stations. Supportive sub-systems (e.g., conveyors and robots) are responsible to implement the transportation task.

In theory, a workflow orchestration subsystem can be integrated in an APC system, e.g., to implement at-line analytics. However, in most cases, the two domains are delineated, and the systems are built by distinct paradigms and using different software tools.

4.3 The LAPP Reference Architecture Model

To answer the need for a harmonized reference architecture model for life science laboratory orchestration, as a part of the Laboratory Automation Plug and Play (LAPP) Reference Architecture Model (RAM), I introduce a multilevel system architecture concept. I do this by mapping the layers of the semantic hierarchical decomposition of chapter 2 to the technical layers of the control architecture.

The control system landscape in laboratory automation is heterogeneous. The functionalities of each type of software system often overlap and depend highly on the specific implementation. To outline a harmonized framework, it is necessary to define each type of software component based on its most common interpretation and usage.

The flow of information in such a system can serve several purposes. The ultimate goal in any laboratory is to generate data about a specific process or material, be it chemical or biological nature. The flow of information in this case begins with the source of the data: the sensor. From here, low-level (embedded) processors (e.g., microcontrollers or PLCs) may pre-process and pass the data along. In many cases, a Device-level Control Unit (DCU) may group additional data streams (e.g., other sensors) together and contextualize the data. These are device-specific processing units, such as chromatography data systems. Above this layer, an Electronic Lab Notebook (ELN) may aggregate data from multiple sources and store it in the context of the sample and/or the process. Overlapping in functionality with ELNs, but implying a broader scope of data management are Laboratory Information Management Systems (LIMSs). In the age of cloud-based big

data architectures, such LIMSs are often interfaced with large-scale company-wide data infrastructures, such as data lakes.

In addition to the acquisition of meaningful data, the integration of laboratory instruments also serves their orchestration and control. In the case of orchestrated workflows, the trigger for an experiment usually originates from a superordinate control layer. This generally corresponds to enterprise-level information systems. On the level of a facility, this order is processed by an orchestrator, which breaks down the order to individual single-laboratory assays. On the laboratory level, a Laboratory Execution System (LES) is responsible for executing the assay on a set of devices, interfacing with their DCUs. In the context of a device, the execution is further broken down to low-level Embedded Controllers (ECs), which ultimately interfaces different actuators and sensors. In addition to detecting primary values of interest (the determination of which is the purpose of the experiment), sensors also deliver values necessary for closed-loop control circles at various levels.

At the low-level side of the subsystems, these control circles may serve the purpose of supportive components, such as pipetting or sample transportation robots. Considering the latter example, let us discuss a multi-level control system. As part of a procedure (assay or experiment), supportive robotic tasks may be necessary, such as the transportation of labware. In this case, the LES would trigger the T task-level RAR for LabwareTransfer. Processing this request, the DCU / robotic middleware breaks it down into S subtasks. On the level of the EC, subtasks are further broken down into Q motion sequences. The elements of these sequences, the P motion primitives represent simple movements between two positions. To complete a motion, the EC must calculate the appropriate trajectory, the velocity- and acceleration profile, and the corresponding forces or torques. Ultimately, a torque is exerted on the robot's joint by setting the appropriate current, e.g., via Pulse Width Modulation (PWM) over an H-Bridge circuit.

The principles of this kind of control cascade can be analogously observed in the example of liquid handling, where the desired T task might be LiquidTransfer, parameterized by the desired volume and the source and destination containers. The DCU would break this down into S subtasks, and trigger the necessary Q motion sequences of the liquid handling arm of the pipettor. On the same level, as the position control of the robotic components, the pipetting function to aspirate the desired amount of liquid is executed by a dedicated pump. In that case, instead of velocity control to achieve a desired position, we are speaking about pressure control to achieve a desired volume. Just as a positon control loop incorporates a velocity control loop, a loop for volume control may incorporate a flow rate controller.

It can be seen that control systems in laboratory automation feature a multi-layered architecture. As such, an analogy can be drawn with industrial control systems. In industrial automation, the business and logistics management function is handled by the ERP system. Analogously, in laboratory automation, high-level data aggregation, LIMS and ELN can be mentioned. Manufacturing operation planning, implemented by the MES, can be interpreted in laboratory automation as the orchestrator, whereas monitoring and supervising is implemented by the scheduler. The control level in laboratory automation is implemented by the DCU and EC layers. The closest to the physical process in both cases is the level with the sensors and actuators.

4.3.1 Layers of the control architecture

Based on the analogy with industrial control systems, I define the following layers of the LAPP control architecture, while using the uniform numbering, introduced in 2.2.1:

7) Lab management (LIMS, ELN) — LIMS and ELN provide centralized digital platforms for managing laboratory data, workflows, and documentation. They support sample tracking, results management, compliance with regulatory requirements, and integration with automation and analytical instruments.

6) Automation scheduler (LES) — An LES is a software layer that orchestrates automated and manual laboratory workflows. It schedules tasks, guides operators through procedures, ensures compliance with Standard Operating Procedures (SOPs), and can interface with both LIMS/ELN and lower-level automation controllers.

3)–5) Device-level control unit (DCU) — The DCU is responsible for direct control and monitoring of individual laboratory instruments, robotic systems, or subsystems. It interprets high-level execution commands, translates them into device-specific instructions, and provides status feedback to higher-level systems such as the LES.

1)–2) Embedded Controller — The Embedded Controller (EC) layer represents the embedded control system that operates at the lowest level of the automation hierarchy. It manages the real-time control of actuators, sensors, and safety interlocks, ensuring precise execution of mechanical, electrical, and fluidic operations within the device or instrument.

4.3.2 Protocol and program layer hierarchy in the LAPP framework

In the section 2.2 of chapter 2, I introduced generic and domain-specific workflow representation frameworks, as well as robotic activity representation approaches. Building on these foundations, I proposed the LAPP RARs—a hierarchical workflow decomposition concept with generalized layers of granularity. In section 4.3.1, within this chapter I defined the corresponding layers of the control architecture.

Bridging the RARs and the execution framework requires protocol and program layers, which encode them in executable or human-readable formats for implementation at the appropriate control levels. Although implementing the full technical stack of protocols and languages for automated laboratory workflows lies outside the primary scope of this chapter and the LAPP framework, it is nonetheless useful to outline a vertical hierarchy of layers, from high-level orchestration down to hardware execution. Here, I recommend how existing building blocks can be aligned with and applied to the conceptual layers of LAPP, noting that boundaries between layers are often fluid and that many state-of-the-art protocols span multiple levels in practice.

Service Protocol (SP) The SP defines the high-level capabilities a laboratory can provide. In this context, a "laboratory" may refer to an organizational unit in academia or the pharmaceutical industry, typically within a single facility and comprising one or more rooms, a range of instruments, supportive equipment, dedicated work areas, and skilled personnel. These services are tied to specific experimental or analytical goals such as

drug discovery, development, or other life science applications. Increasingly, *cloud laboratories* offer these capabilities remotely, enabling internal or external clients to submit samples along with instructions specifying reagents, buffers, and the required processes. The SP, as a conceptual layer, formalizes such offerings—for example, high-throughput screening, HPLC analytics, or next-generation sequencing. In practical terms, this concept is exemplified by commercial platforms such as the *Emerald Cloud Lab*, where users design and dispatch high-level experiments via the *Command Center*. These workflows are encoded in the Symbolic Lab Language (SLL), a domain-specific, machine-readable language that abstracts experimental operations into standardized symbolic instructions [117]. SLL enables complex experimental protocols to be represented in a reproducible, platform-independent form, while still being fully executable by ECL's automation infrastructure [118]. This way, SLL spans from SP through Experiment Design Language (EDL), all the way down to Laboratory Process Language (LPL) - as defined below.

Experiment Design Language (EDL) The EDL is an outcome-focused recipe at the workflow or assay level, answering the question *"what to do"*. It defines the sequence of experimental steps—such as sample preparation, incubation, stirring, or temperature control—independently of specific devices. Because it abstracts from hardware, an EDL protocol can be transferred between laboratories with similar capabilities. Examples include Laboratory Open Protocol (LabOP) and Chemical Description Language (XDL), as discussed in 2.1.1.

Laboratory Process Language (LPL) The LPL translates the "what" from the EDL into the "how" of execution for a specific laboratory configuration. It defines the sequence of commands, parameters, and device-specific instructions that can be executed on the actual available instruments. Although some generic preparatory and diagnostic LPL frameworks exist, most system integrators and software providers implement device-specific drivers at this level due to the wide diversity of instruments. The LAPP framework places particular emphasis on supportive equipment, such as robotic sample transport systems, from this layer downwards.

Modular Robot Program (MRP) The MRP bridges the LPL and the robot-specific Low-level Robot Program (LRP). It contains reusable and parameterized motion sequences, organized into Ttasks and Ssubtasks. An MRP can, for example, be implemented within a SiLA 2 server, where each SiLA command invokes a corresponding low-level robot routine. This modular design promotes reusability, simplifies orchestration, and allows robotics functions to be integrated seamlessly into broader laboratory workflows.

Low-level Robot Program (LRP) The LRP encodes primitive robot motions and simple IO logic. It is typically written in a vendor-specific programming environment and executed directly on the robot controller (e.g., ABB RobotStudio, UR PolyScope, KUKA KRL). Each motion primitive defined at this layer must ultimately be translated into joint-level commands.

Joint trajectories and IO (JTIO) At the lowest layer, the robot controller converts Cartesian waypoints into joint angles using inverse kinematics, interpolates the motion between these points, and executes the trajectories in coordination with input/output control. This hardware-level execution ensures that all higher-level instructions—from the MRP and LRP—are realized accurately and consistently in the physical robot's movements.

In table 4.2 I map the three hierarchical aspects of LAPP-RAM to each other. It is important to mention that the borders between the levels are not strict, and some implementations may fuse or bridge certain levels entirely. There are strong correlations between the three dimensions, meaning, for example, that a certain level of process activity is usually represented by a certain type of protocol and is executed on a certain level of the automation system. However, these relations are not strict either. Depending on various factors, such as the size of the laboratory and the existing boundary conditions of the infrastructure, implementations may vary in each aspect.

TABLE 4.2
THE THREE HIERARCHICAL ASPECTS OF LAPP-RAM, MAPPED AGAINST EACH OTHER

Level	Process	Protocol	Technical
8			
7	Service	Service pr.	LIMS, ELN
6	Procedure	EDL	LES
5	**Task**	LPL	DCU
4	**Subtask**		
3	**Motion sq.**	MRP	
2	**Motion prim.**	LRP	EC
1	**Actuator pr.**	JTIO	
0			

4.3.3 Integration of the DT

To facilitate the APC or orchestration, a DT framework must be closely integrated with the control architecture. As such, a scheduler must be able to query a device's DT for the device's semantic capability descriptors, in order to instantiate it and make its commands available to be used in the LPL. In the case of a transportation robot, the robotic middleware (DCU) must fetch the geometrical information (i.e., positions and labware definitions) from the DT of each device, in order to parameterize the motions accordingly. When a DT framework is incorporated into the appropriate levels of the control architecture, the need for manual configuration and robot teaching can be minimized or eliminated.

4.3.4 Applicability of the LAPP-RAM

As discussed in section 4.1 at the beginning of this chapter, the LAPP RAM was developed with three main objectives:

1. to define the key system components, their relationships, and integration principles;
2. to delineate clear roles and responsibilities between the layers of the control architecture; and

3. to unify the previously introduced concepts of the LAPP framework by mapping the semantic workflow decomposition (RARs) onto the generalized layers of the control architecture.

By doing so, I provide a blueprint and best-practice guidance for creating a scalable, modular ecosystem within the multi-layer, heterogeneous landscape of laboratory automation.

Without such a reference architecture, implementations often result in heterogeneous ecosystems where boundaries between components and layers are poorly defined or sub-optimally allocated. This is evident in some commercially available platforms where critical integration logic resides entirely with the system integrator's software stack. In such cases, typically the same vendor is responsible for multiple vertical integration layers, and interfaces are proprietary, preventing modular reconfiguration or replacement of individual components. This leads to vendor lock-in, limits flexibility, and impedes scalability.

In contrast, when the LAPP-RAM principles are followed, each layer of the control architecture is implemented with clear separation of concerns, standardized interfaces, and well-defined responsibilities. The result is a modular ecosystem where devices, software services, and orchestration layers can be integrated, replaced, or scaled independently.

The value of this approach is illustrated in the reference implementations presented in chapter 5. Despite significant differences between the systems—a ROS-based academic prototype mobile manipulator and an industrial implementation by a commercial integrator—both demonstrate adherence to the same set of agnostic principles defined by the LAPP-RAM. Figure 5.5 and figure5.7 in particular show how the layers of the LAPP-RAM are concretely realized in both cases, providing direct examples of how this agnostic framework can be applied in practice.

4.4 New scientific results

Thesis 3

I created a generic architecture model (Laboratory Automation Plug and Play (LAPP)-Reference Architecture Model (RAM)) for the integration and automation of equipment in life science laboratories. I demonstrated its applicability on the basis of a prototype and an industrial pilot implementation (see chapter 5).

I showed that hierarchical function-centric models can be combined in a single generic architecture model (LAPP-RAM) by mapping the layers of the semantic decomposition to the layers of the control architecture, as shown in table 4.3. The table follows the consistent leveling scheme used throughout the thesis, which is why the top row is intentionally left blank.

The protocols are:

- Service Protocol (SP)
- Experiment Design Language (EDL)
- Laboratory Process Language (LPL)
- Modular Robot Program (MRP)
- Low-level Robot Program (LRP)

- Joint trajectories and IO (JTIO)

The technical layers are:

- Laboratory Information Management System (LIMS)
- Electronic Lab Notebook (ELN)
- Laboratory Execution System (LES)
- Device-level Control Unit (DCU)
- Embedded Controller (EC)

TABLE 4.3
THE THREE HIERARCHICAL ASPECTS OF LAPP-RAM, MAPPED AGAINST EACH OTHER

Level	Process	Protocol	Technical
8			
7	Service	SP	LIMS, ELN
6	Procedure	EDL	LES
5	**Task**	LPL	DCU
4	**Subtask**		
3	**Motion sq.**	MRP	
2	**Motion prim.**	LRP	EC
1	**Actuator pr.**	JTIO	
0			

Related publications: [WA2, WA3, WA5, WA6, WA7]

Part 5

IMPLEMENTATION

5.1 Motivation

This chapter presents an experimental demonstration and validation of the concepts that I introduced in chapters 2, 3, and 4. I demonstrate the feasibility of two of the main conceptual pillars. First, a research-oriented platform was used to create a prototype, followed by a pilot with an industrial-grade mobile manipulator.

The first preliminary step was to manifest the semantics of chapter 2 in the form of a service description, more specifically in the form of a Standardisation in Laboratory Automation Consortium (SiLA) feature definition.

The academic prototype implementation was conducted using a research-oriented Mobile Manipulator robot (MoMa) platform, TIAGo ++ [45]. Based on Robot Operating System (ROS), the software stack was adapted to reflect the layers of the Laboratory Automation Plug and Play (LAPP) Reference Architecture Model (RAM) (as presented in chapter 4). Implementing a SiLA-ROS bridge, the labware transfer functionality was exposed towards a SiLA based scheduler. This study served as a prerequisite for the next phase.

As the third step, a market-ready implementation followed with an industrial-grade MoMa of EngRoTec, called mobERT [119]. First, the hardware was adapted to the labware transfer task by developing an end effector for microplate manipulation. Furthermore, the robot was equipped with an on-board storage (hotel), and a tool changer to enable adopting to further types of labware. Based on the LAPP semantics, the `Labware-␣Transfer` sequence was implemented and exposed via a SiLA driver. This interface was implemented as a wrapper on top of EngRoTec's middleware/fleet manager ERTmiral. The control architecture complies with the concept outlined in chapter 4, by keeping the high-level role of the scheduler, using an interoperability protocol to trigger the T task- and S subtask-level commands, and a middleware breaking these down into low-level functionalities, such as Q motion sequences and P motion primitives.

5.2 Preliminary work in the SiLA Robotics Working Group

As discussed in section 1.4, standardized interoperability of laboratory devices and supportive robotics is essential to enable the integration of these components into overarching

automation ecosystems. SiLA and Laboratory and Analytical Device Standard (LADS) can be named the two most prominent industry-wide initiatives in this regard.

As a part of the SiLA consortium, the SiLA Robotics Working Group (SRWG) [1] constitute a domain-specific group of Subject Matter Experts (SMEs). The SRWG's key work packages include the assessment of user needs in the labs, and identifying present and future robotic capabilities that can answer these needs. We published the results of this endeavor in [WA5]. We identified the labware transfer capability as the most requested item. To answer this need, a broad range of state-of-the-art fixed base and mobile robotic solutions are already available on the market. We also condensed down the additional robotic capabilities that are not yet addressed by mature off-the-shelf robotic solutions. As a result, a categorized system of namespaces and the stubs for each activity were created [85].

The next step by the SRWG was to unify the feature definition for labware transportation. It has been made available on the SiLA GitLab site [84].

The features were divided into two sets: one for devices that are used for manipulating objects (called *LabwareTrasnfer**Manipulator**Controller*); and one for passive lab equipment (called *LabwareTransfer**Site**Controller*). Other design principles during definition creation were:

- to be device-agnostic, i.e., to be compatible with any robot (fixed base or mobile), conveyor, or other device with the same functionalities, regardless of the manufacturer,
- to be generally vendor agnostic, so the usage does not depend on the manufacturer,
- to be modular with distinct categories,
- to represent the Ⓢ subtask level Robotic Activity Representations (RARs) in the form of commands.

After the feature definition has been unified, the SRWG created reference implementations for stationary robots, as a part of the BioLAGO SiLA 2 AniML Serial Hackathon (BioSASH). This was a series of hackathons co-organized by BioLAGO [120] and the SiLA Consortium. These events gave people from different fields the opportunity to work together in hopes of finding new solutions to solve common challenges in process automation.

During the third installment of this series, the goal was set out to the SiLA-based communication to the robots' driver software. The result can be found on GitLab [121] as the SiLA-ROS bridge. This solution is further discussed in Section 5.3.6.

The fourth installment of BioSASH took place at the end of September 2021 in Konstanz, Germany. One of the five working groups set the goal to "SiLA-fy" the labware transportation process for stationary robots [122]. Also presented as part of the event was a general example implementation of the low-level side of the labware transportation activity. As a basis, the group used the `LabwareTransferManipulator-Controller` and `LabwareTransferSite-Controller` feature definitions [123]. A working implementation was delivered for the Universal Robots UR3 and a PreciseFlex 3400 robotic arm, performing a pick-and-place task from a source hotel to a destination

[1] The SRWG is lead by Ádám Wolf since March 2022 and as of the time of writing this present thesis.

hotel, passing the plate from one robot to the other in-between through a shared handover site. This implementation was the first demonstration of the *unified feature definition* for labware transfer tasks, and the learnings were incorporated in the final version. The results can be reached as a GitLab project [124] and the final presentation, given at the end of the event, can be reached here [125].

These steps constitute prerequisites for creating implementations of the LAPP concept for ROS-based MoMas using the SiLA feature definitions and the SiLA-ROS bridge (see more details on the latter in section 5.3.6).

5.3 Prototype implementation

The purpose of this preliminary study was to demonstrate the possible usage of the LAPP model on a MoMa in a real-world setting. We carried out the reference implementation on a PAL Robotics TIAGo++ unit (from now on, *TIAGo* or *robot*). The project aimed to implement a pick-and-place laboratoryware transfer functionality between two stations for microplates[2].

5.3.1 Experimental setup

The prototype implementation is based on the TIAGo++ mobile manipulator interacting with two experimental stations (Station A and Station B). Each station is equipped with a mechanical fixture for holding microplates and a fiducial marker for visual localization. The robot is fitted with custom 3D-printed gripper fingers, as well as a front-mounted microplate holder ("hotel") for safe transport of labware between the stations (Figure 5.1).

The layout of the stations is illustrated in Figure 5.2, which provides a top-down schematic of a single site. The diagram shows the position of the point of interest (PoI) on the floor, the fiducial marker, and the designated handover site for the manipulated object, together with the associated coordinate frames. This representation clarifies how the robot localizes and approaches a station during the labware transfer workflow.

An overall view of the prototype in operation is given in Figure 5.3. It combines three perspectives: the robot's head camera image, the RViz visualization of the navigation map and robot state, and an external view of the TIAGo++ interacting with the stations. Together, these perspectives illustrate how the system perceives its environment, plans navigation, and performs the microplate transfer task.

The workflow demonstrated in the prototype consists of a complete labware transfer cycle. The robot first localizes the fiducial marker at a station and positions itself at the corresponding PoI. Using the custom gripper, a microplate is picked from the fixture and placed into the front-mounted holder for safe transport. The robot then navigates to the second station, where the process is repeated in reverse: the plate is retrieved from the holder and placed into the designated fixture. This sequence illustrates the end-to-end execution of the labware transfer task, serving as a practical validation of the prototype setup. The complete operation consists of the following generalized steps (see more detail in chapter 2):

[2]Meets the Standards ANSI/SLAS 1-2004 through ANSI/SLAS 4-2004.

Fig. 5.1. Images of Station A (1) and Station B (2) and the custom-made gripper, both in open (3) and closed (4) positions

- Undock from Docking Station
- Drive to Point of Interest (PoI) Station A
- Pick up the object from Station A object holder site
- Place the object to the onboard holder
- Drive to PoI Station B
- Place the object to Station B object holder site
- Drive to Docking Station PoI
- Dock, if charging is needed

5.3.2 System architecture

The proposed system architecture is based on the integration of the ROS and SiLA frameworks. Figure 5.4 describes the connection between the ROS and SiLA ecosystems at the level of the main components. The contract for communication between the SiLA server and SiLA client is the *feature definition*. The bridge represents the SiLA-ROS bridge

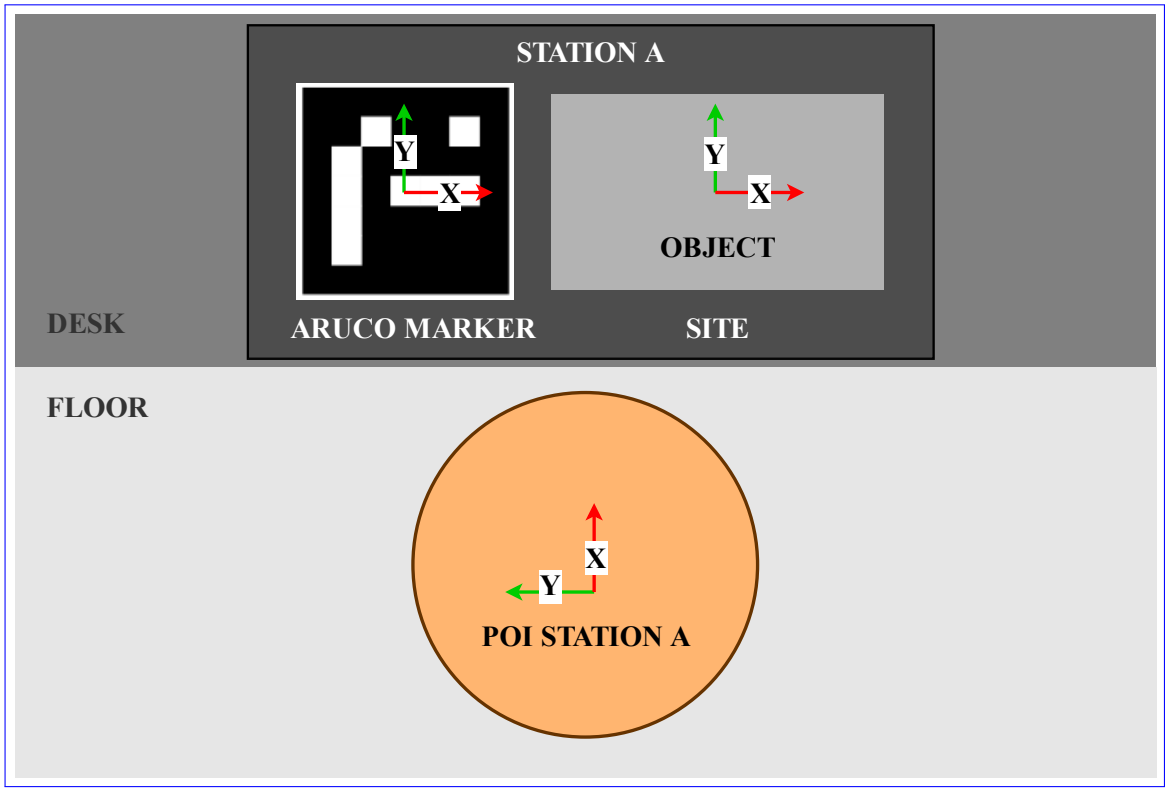


Fig. 5.2. Close-up diagram of a station. The diagram represents a top-down view of a station, including a POI position on the floor (yellow circle), the dummy device itself on top of the desk (dark grey), including a marker and a site (light grey), where the manipulated object can be placed. The coordinate frame at the POI represents the base of the robot, in the correct direction, when it is doing the manipulation. The frame seen on the marker is placed in the direction as the robot's head camera sees it. The frame of the site is expressed as a translation in X and Y, relative to the marker's frame. All three frames have red axes for X, and green for Y.

package [126], which is based on a reference implementation published on the SiLA GitLab [121]. On the ROS side, communication is managed through the standard concepts: topics, services, and actions. Further descriptions of SiLA client, SiLA-ROS bridge, ROS action server, and feature implementation components are detailed in 5.3.6.

Two main ROS nodes have been implemented: one handles the ArUco marker-based arm control on the low level, while the other one is responsible for the high-level execution of the transportation sequence. These two nodes, respectively, represent an action server written in C++ and an action client which is a Python program. Both are embedded as ROS nodes into the robot's functionalities which are initiated at startup automatically. A detailed explanation will follow in Sections 5.3.4 and 5.3.5.

In Figure 5.5, a layered view of the system architecture can be seen, using the numbering that I introduced in chapter 4. The layers span from the low-level actuator and sensor control through the embedded ROS, and device-level controllers (SiLA) to the automation scheduler Universal SiLA client (USC). On the device-level control layer, the Digital Twin (DT) of a generalized (dummy) device is depicted. In detail, I presented the canonical layered architecture in chapter 4. The representation here is an example of its implementation, using specific technological building blocks.

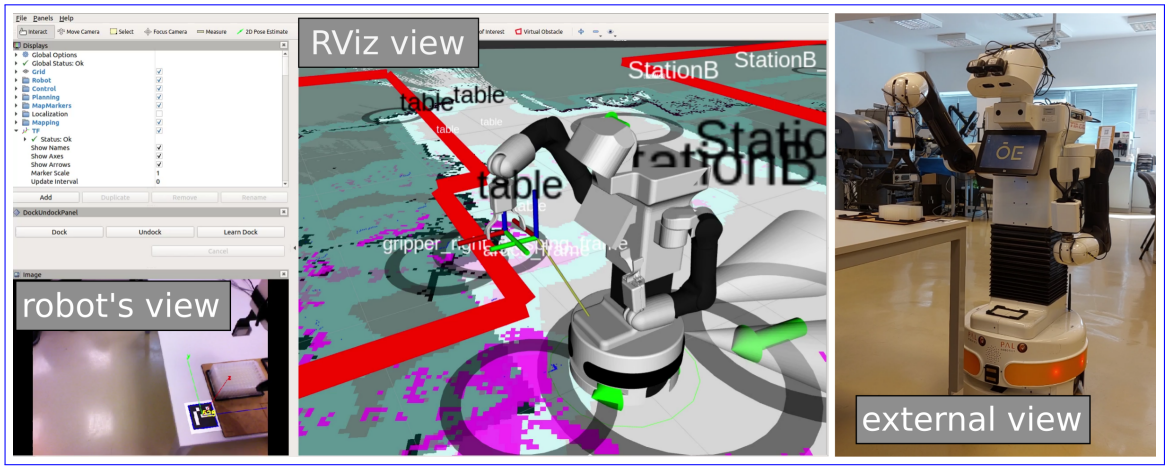


Fig. 5.3. Testing of the ArUco-based arm motions and the microplate picking, including the robot's head camera view (left) the RViz view of the simulated robot state (middle), and an external view of the robot (right). In the RViz view, dark grey areas framed by red lines are restricted areas, light grey areas show where the robot is allowed to navigate, the green arrows are the POIs placed on the map, magenta dots are the real-time obstacle data detected by the LIDAR, grey circles mark the virtual objects on the map which can be modified by dragging and moving them

5.3.3 Technology mapping

To map the relevant ROS and SiLA functionalities to the LAPP concept, we created a Unified Modeling Language (UML)-like diagram, seen in Figure 5.6. We created this representation with different levels of RARs in relation to each other with color and letter coding. Texts in blue are depictions of SiLA elements (features, commands, or attributes), while yellow text represents an implementation as ROS functions (actions, services, and topics), all in correlation to the LAPP representation. The UML namespaces represent a categorization of RARs. The Ⓣ Task- and Ⓢ Subtask-level RARs represent outcome-oriented activities, which we categorize based on the function. In contrast, Ⓠ Motion sequence, Ⓟ Motion primitive, and Ⓐ Actuator primitive RARs are considered to be low-level activities, which are, in most cases, masked away from the scheduler. We categorized these according to the robot components. In [WA5] presented a structure for the extended taxonomy of outcome-oriented RARs [85].

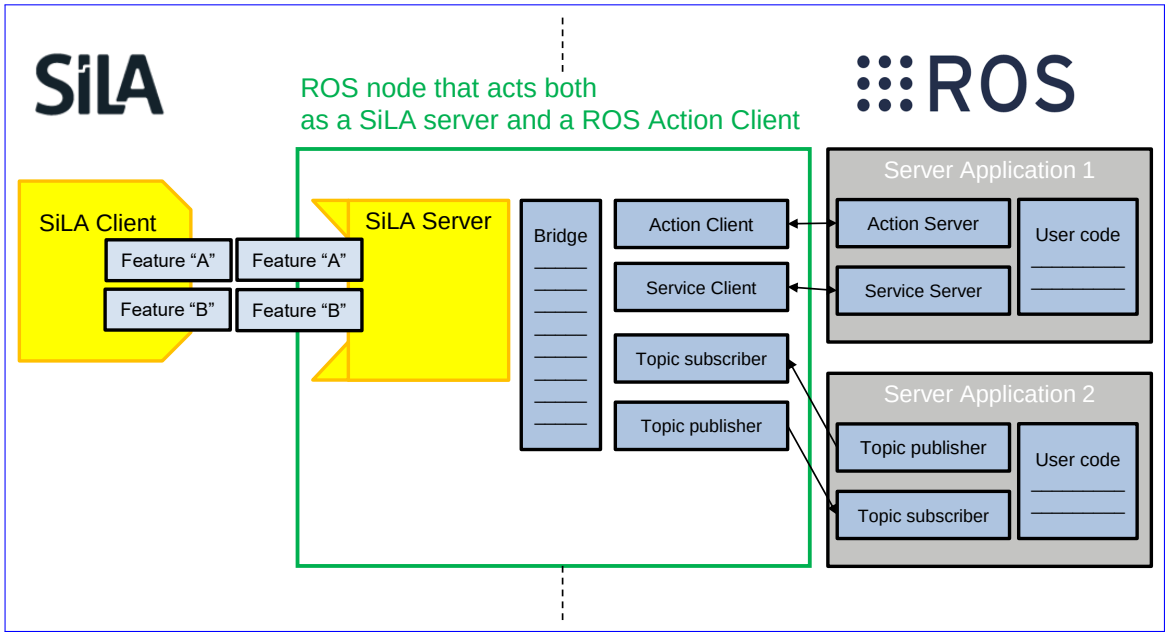


Fig. 5.4. Conceptual software architecture of the SiLA-ROS hybrid system.

A detailed sequence diagram of the labware transfer process between a fridge and a pipettor device is presented in the Supplementary material (S3).

5.3.4 Action server implementation

As outlined in Section 5.3.3, we implemented the Arm Motion Sequence Q `MoveThrough-Sequence` in the form of a ROS action server, written in C++. The name of the node is `ArucoMotions`, referring to the functionality it accomplishes. This action server is started during the robot's startup process with the other default startup nodes.

The node handles the detection of the ArUco marker and the derivation of the site (s) and the site approach (ss) positions as transformations from the marker's frame. It also executes a MoveIt [127] based motion to move the Tool Center Point (TCP) to those frames. The two dedicated frames are directly above the microplate holder slot at two different distances from the plate itself. (ss) is used for approaching, while (s) is for the final position which is used when grabbing the plate. Both are expressed relative to the marker, with translations along the x, y, and z axes.

When the upcoming step of the overall pick-and-place sequence is an ArUco-based motion, this node checks if there is a marker on the head camera's image. If not, it throws an error, if there is one, it calculates the transformed frames and sends a goal position through the C++ MoveIt interface. While this executes, the rest of the main sequence (seen in Section 5.3.5) is blocked. Every time the main sequence calls this node and there is a marker in sight, the motion starts.

Using the naming conventions of [WA4], the coordinate frames are defined as follows. These positions are to be fetched from the DT of the respective (dummy) device, as proposed in chapter 3.

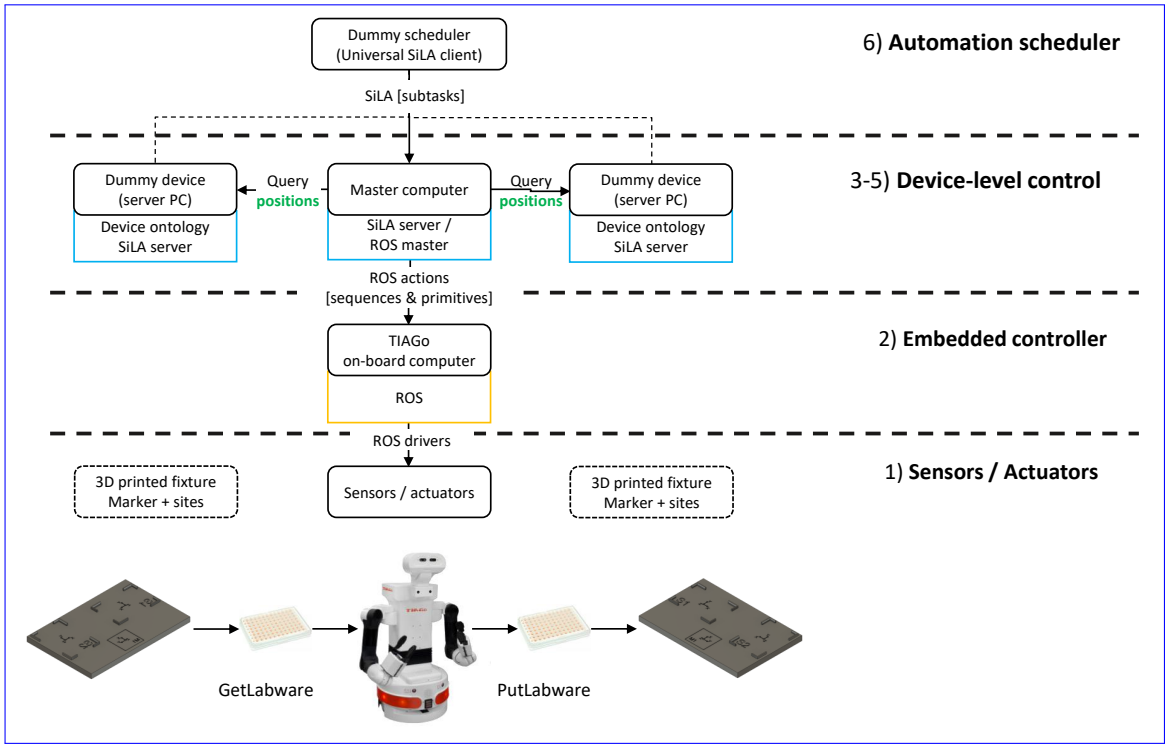


Fig. 5.5. Architecture of the prototype implementation by layers: 1) Sensors/Actuators layer, 2) Embedded controller layer, 3-5) Device-level control layers, 6) Automation scheduler layer. The ROS component is outlined with yellow, and the SiLA components with blue.

- PoI STATION A (PoI) - base pose for the station
- ARUCO MARKER (m) - fiducial marker for device
- SITE (s) - handover site (plate nest) of device
- The site approach (ss) and device approach (sd) frames are the same in this instance since a device includes only one site.

While the prototype implementation was designed to accommodate the digital twin aspect of the LAPP concept by leveraging the ROS ecosystem together with the coordinate transformation framework `tf`, the final setup did not fully implement this functionality. Instead, the positions were determined relative to the marker, which effectively represents the initial creation of a digital twin prototype of the dummy device. However, these positions were not integrated into a comprehensive digital twin instance; rather, they remained embedded in the sequence implementation, as described in Section 5.3.5.

5.3.5 The Sequence

We implemented the sequence in the form of a ROS action client. The node `transportation(transportationScript.py)` handles the sequence steps as a Python script. In [WA4] we introduced a generalized sequence, for which a detailed implementation is presented in this thesis.

In the supplementary table S4 we present how specific ROS-based functions implement each RAR in the canonical sequence. Moreover, we list the digital twin parameters

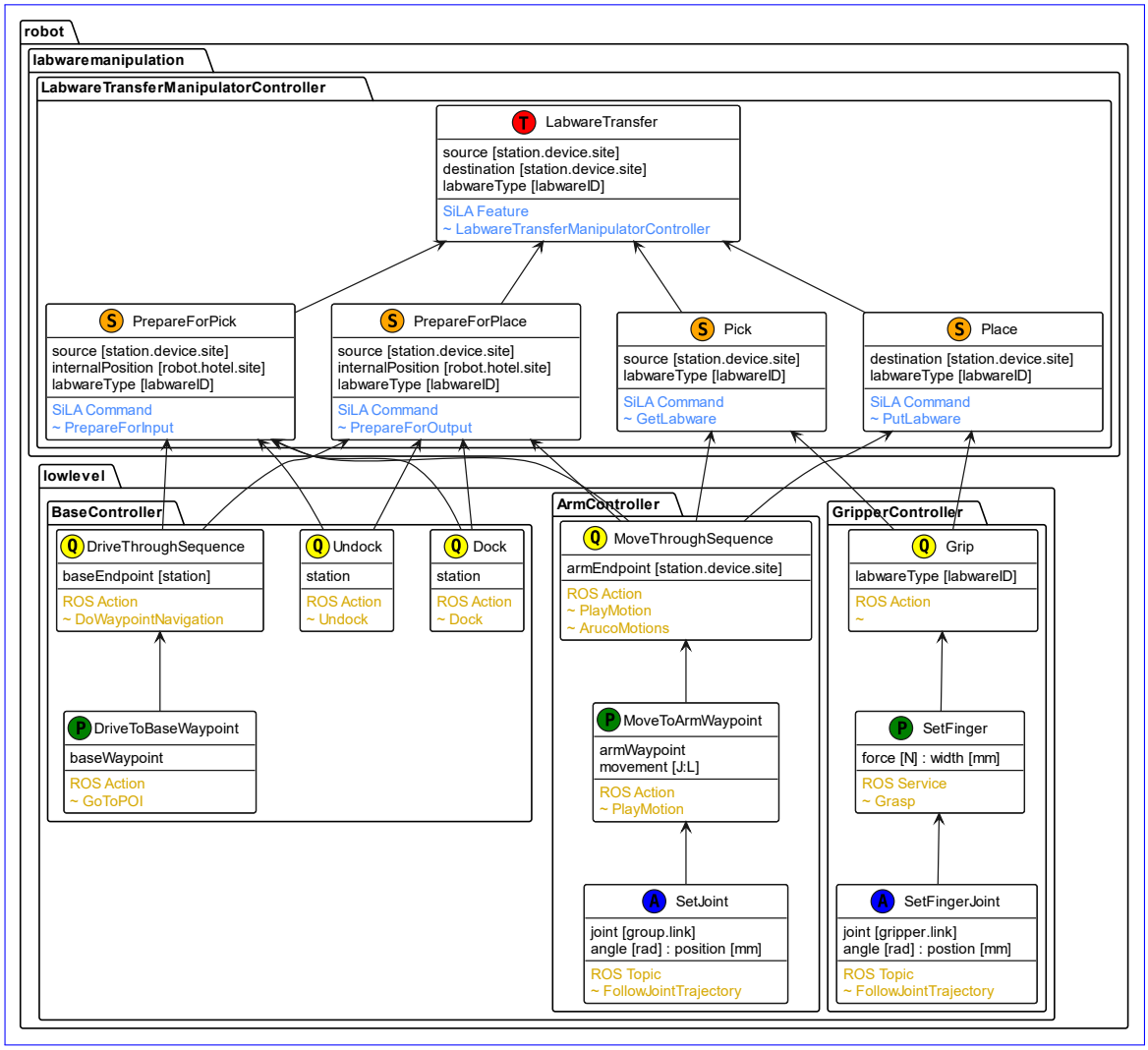


Fig. 5.6. UML-like representation of the labware transfer RARs for the prototype implementation. Namespaces represent a way of categorizing the RARs. The arrows represent how a high-level RAR is composed (broken down into) low-level RARs. Task (T), Subtask (S), Motion sequence (Q), Motion primitive (P) and Actuator primitive (A) are represented as classes. Parameters are shown as class members, with the data type in square brackets []. Overloaded parameters (in the case of polymorphic RARs) are separated by a colon (:). Mapping the SiLA implementation is set in blue font, while the ROS implementation is in yellow. The names of the specific ROS functionalities are marked by a tilde (∼)

for each function. In a LAPP-compliant system, these parameters are represented in the digital twin instance of the corresponding components.

5.3.6 SiLA-ROS bridge

A crucial part of the implemented structure is the SiLA-ROS bridge. This allows for smooth communication between the ROS and the SiLA ecosystems, based on a client-server model. It helps to connect one or multiple ROS-based devices at once to the higher-level SiLA control layer. In the open-source software repository of the SiLA Consortium exists a reference implementation for realizing the SiLA-ROS integration [126]. During

our prototype development project, we encountered problems with version compatibility of ROS, Python, OS, and robot firmware, making it difficult to find a configuration where all components could communicate. Attempts to run the Python 3-based SiLA-ROS bridge on the robot with ROS Melodic on Ubuntu 18.04 proved unfeasible. However, such difficulties can be avoided in the future, as each subsystem is continuously maintained.

Let us consider the control cascade across the layers of the architecture. The command call originates from the SiLA client in the direction of the SiLA server. A *command* represents a RAR (presented in Figure 5.6). Then, the server calls the RAR function on the feature implementation's side, which registers callbacks toward the bridge. The bridge breaks down the callbacks into ROS functionalities by calling the respective ROS actions on the ROS action server. When the action ends, the result is sent back all the way to the SiLA client through the respective interfaces. From the first call and the succeeded state, continuous feedback is given among all interfaces.

5.4 Experimental results

The efforts connected to the LAPP initiative span across the *SiLA Robotics Working Group*'s activities and the prototype development project presented in this chapter. The different milestones of this initiative can be assigned to the Technology Readiness Level (TRL) scale [128], these levels can help when describing the testing status.

Testing took place in two phases, at two different locations. The first phase was at the Antal Bejczy Center for Intelligent Robotics (ABC-iRob), while the second was at a pharmaceutical company's development laboratory. As for the first phase, we created a TRL 4 test scenario for the pick-and-place testing, while the second phase concluded a TRL level 5 scenario, in a relevant environment, in this case a pharmaceutical laboratory. Both phases verified the same pick-and-place scenario with as similar setups as possible in order to create consistent and repeatable environments.

5.4.1 Test runs

We tested different Ⓢ Subtask-level sequences multiple times, namely the navigation, the picking and placing separately and the Ⓣ Task-level sequence as a whole. These activities showed different levels of robustness during testing.

At first, we focused on testing the `ArucoMotion` node, while standing in one place in front of a station, starting from the *Benchtop* configuration, looking down at the station. We ran the following steps in a loop:

- Open gripper
- Move to the site approach position, based on marker transformations
- Move down to the site's level
- Grasping
- Move up to site approach position
- Move back down to the site's level
- Open gripper

- Move up to site approach position

This sub-sequence was running for around ten minutes, when one of the gripper's motors overheated, generating an error. This caused one of the fingers to get stuck in one position. This way, testing could not be continued, only after cooling the motor back to a usable temperature. This was caused by a bug in the grasping functionality, which uses force feedback. Apart from this problem, the accuracy of the gripper placement and arm movements proved to be adequate. The repeatability and reliability proved to be insufficient, mostly during the placing of the microplate on the site. Around 10% of the placements were not accurate in that the plate was misplaced by a few millimeters. This caused the plate to not fit correctly in the socket. The most obvious solution would be to redesign the site sockets to be more forgiving and have a greater zone of success. The inaccuracy is an accumulation of mechanical play in the kinematic chain and the error of the vision system. Addressing these limitations lies outside the scope of the present thesis.

Next, we tested the navigation with the simple steps of sending the robot to the three PoIs repeatedly. Out of the three subtasks, we observed the biggest inaccuracy in this step. Imperfections of the floors in both locations were causing troubles for the robot due to its six-wheel setup. It got stuck easily, caused by situations where one of the driven wheels were lifted above the floor. This misalignment ultimately disoriented the odometry. In instances when the laser localization cannot correct for the mechanical deviations of the movements, the robot's perceived location and orientation can shift.

Finally, we tested the whole sequence at once. Since the duration of the overall sequence was more than six minutes, and, due to the nature of the disorientation error, it required constant supervision, the number of such complete test runs was limited. Even though there were some successful rounds also recorded on video, most runs failed at some point. See the video of the final test sequence in four times acceleration in the supplementary materials S5. As a solution, the possibility of using a different robot can be helpful to build a more robust system. Another solution is to place ArUco markers or use other localization tools e.g., on the floor where the robot can easily detect them and set fixed PoIs this way.

5.4.2 Results

In summary, our testing showed that the bottleneck lies in navigation inaccuracy, as malfunctions occur when the robot traverses on an uneven surface. The 6-wheel design of the Autonomous Mobile Robot (AMR) proved to be a sub-optimal setup for this application.

Other problems arose, especially in the ABC-iRob laboratory, with the network bandwidth, probably caused by the high demand of image broadcasting via ROS, used for the ArUco marker detection. This constant stream of images seems to be excessive on a shared network where the only connection to any mobile robot is through WiFi or wireless connection.

During this project, the supported software environment consisted of Ubuntu 18.04, ROS Melodic any Python 2, which was already outdated. This resulted in compatibility problems concerning the SiLA implementation since the SiLA environment requires Python 3. According to the PAL Robotics roadmap, TIAGo robots will have a major software update with ROS 2 and Python 3 support.

Despite the difficulties with the hardware, the setup proved to be a sufficient platform for a prototype implementation to demonstrate the LAPP concept.

5.5 Industrial pilot implementation

After the prototype implementation of TRL 4-5 presented in section 5.3, this section demonstrates the real-life (TRL 5-6) implementation of the LAPP concept's service descriptions and RAM. We utilize the unified SiLA `Labware-Transfer` feature definitions [56], relying on the work of the SRWG. We implement a SiLA server for the industrial-grade MoMa of EngRoTec Solutions, called mobERT. We introduce the specific hardware adaptations that equip mobERT to be a laboratory manipulator capable of handling microtiter plates (MTPs). We implement the LAPP RAM in a multi-level system architecture. The scheduler layer is represented by Biosero's GBG [65], the interoperability protocol by SiLA, and the middleware/Device-level Control Unit (DCU) by EngRoTec's ERTmiral. The latter fulfills fleet management functionalities and the integration of the MoMa's components. These include the control subsystem of the autonomous mobile robot and of the collaborative robot (cobot).

To implement the labware transfer capability for a MoMa we relied on the LAPP framework. We utilized the semantic descriptions, which were standardized in the form of the SiLA feature definitions. Furthermore, we set up the orchestration of our laboratory workflow based on the LAPP generalized hierarchical workflow decomposition. We set up the overall system architecture according to the generalized RAM, as shown in figure 5.7. We utilized the learnings of our preliminary academic prototype development feasibility study 5.3. Contrary to the ROS based architecture that we used there, for our industrial-grade implementation, we utilized proprietary low-level control components. With that, we demonstrate that the LAPP framework is modular and agnostic towards low-level implementation. Figure 5.7 displays the system architecture, according to the generic layers, as outlined by the LAPP RAM.

5.5.1 The workflow

The part of the workflow that we set as the scope of our automation endeavor covers the preparation of the samples for subsequent analytics, e.g., for High-performance Liquid Chromatography (HPLC). Our first-stage laboratory setup involved two primary devices. We will refer to these as source and destination devices, which implement the *Labware-Transfer**Site**Controller* feature.

- **Fridge:** A conventional 4 °C fridge, serving as a sample storage unit (Liebherr Premium No-frost).
- **Andrew:** A benchtop pipetting robot, serving as a sample preparation station (Waters Andrew+).

In this chapter, we will discuss the implementation of the T `Labware-Transfer` task, as a part of this workflow. For this, the samples (carried on MTPs) must be picked from the fridge and handed over to Andrew. The latter must then start its sample preparation method.

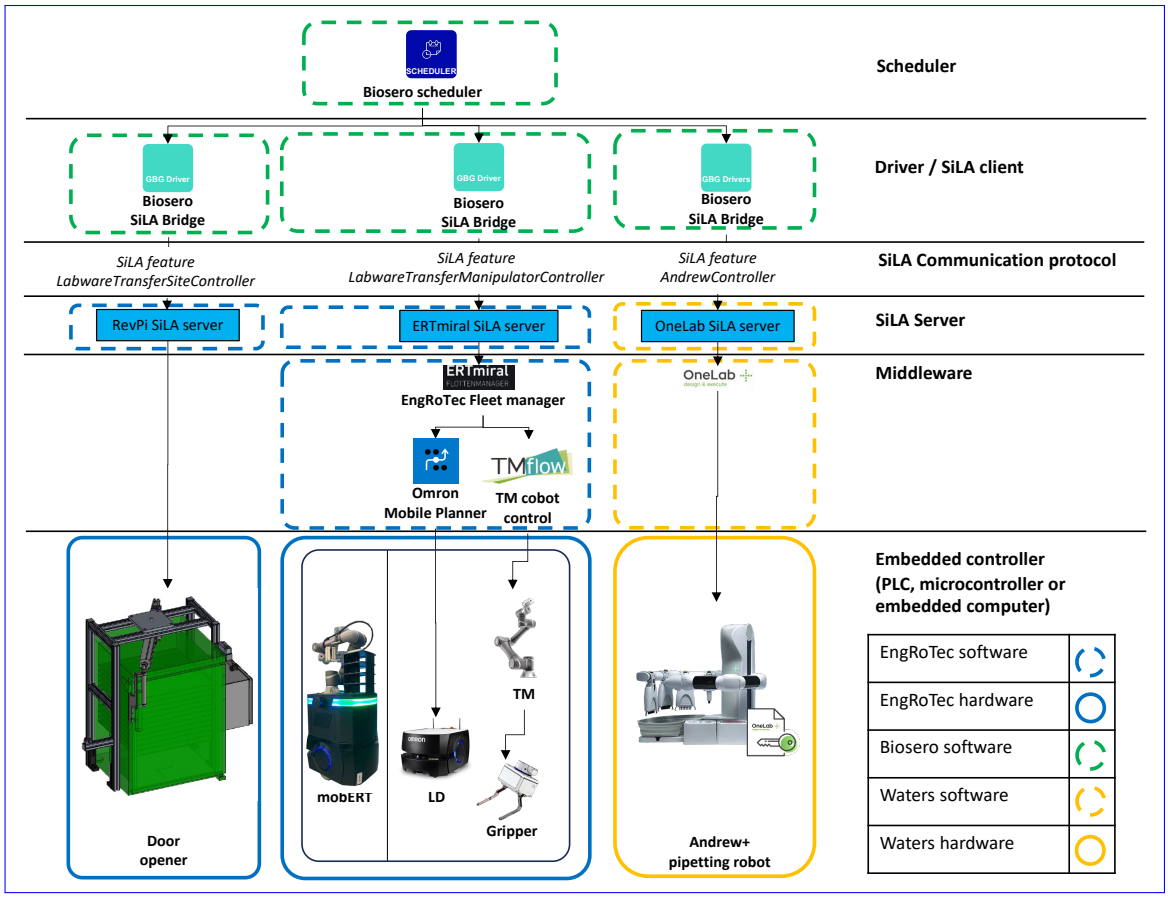


Fig. 5.7. System architecture of the industrial pilot implementation

We picked this workflow as a target for our project because it features elements that are common across a wide scope of laboratory workflows. In a generic view, most laboratory setups involve some kind of sample storage, one or more pipetting stations, and one or more analytical instruments. Incorporating examples of these generic building blocks makes it possible for the resulting pilot setup to be easily adapted to a variety of different sub-domains of pharmaceutical laboratory automation.

5.5.2 The scheduler

To orchestrate the workflow, we were looking for a scalable software solution that is compliant with standardized interoperability. We have identified Biosero's GBG scheduler [65] as a prevalent and proven solution across a wide range of laboratory setups across the pharmaceutical industry. A decisive factor was its compatibility with the SiLA 2 interoperability protocol.

We created a basic GBG method as a sequence of SiLA commands of the `Labware-⌋ Transfer` feature definitions [129]. See section 5.5.8 for more details. This included the control of both primary devices (the fridge and Andrew), via the LabwareTransferSiteController. In the context of a labware transfer, the source and the destination devices must be prepared for the output or input of labware. It has to be ensured that the device is in a suitable state that makes the transfer possible. No methods can be running that would prevent the transfer, and the handover site must be exposed for the external manipulator to handle the labware. This can mean opening a lid or door, such as in the example of the fridge. We deployed the scheduler on the same Industrial PC (IPC) as the robotic middleware (see in section 5.5.6).

5.5.3 The SiLA layer

GBG interfaces with integrated devices by means of drivers. Biosero maintains an extensive library of proprietary device drivers. This means, that they have created wrappers around each of these device's Application Programming Interfaces (APIs), exposing a proprietary communication channel towards GBG. In contrast to that, their generic SiLA driver makes it possible to integrate any device that has a functioning SiLA server. By parsing the corresponding feature definition, GBG populates its list of commands for the device and makes it available for use within a method. For this, the GBG SiLA bridge implements a SiLA server on one end, and a generic GBG device on the other end. As such, it is analogous to the SiLA-ROS bridge that we have presented in [WA6].

The counterpart of every SiLA server is a SiLA client. This implements the service provider's (i.e., the laboratory device's or robot's) high-level control interface. It is either built-in natively to the middleware (running on the DCU) or implemented as a wrapper around the middleware's proprietary API.

5.5.4 The fridge door opener

As the primary source device, our sample storage fridge had to be equipped with an automatic door opener. Implementing the `LabwareTransferSiteController` fea-

ture's PrepareForInput and PrepareForOutput functionalities, the door opener must expose the inner handover positions (sites) for the robot to pick or place the MTPs.

To fulfill this functionality, we built a robust aluminum frame around the fridge, which houses the electric cylinder and the control box. The cylinder connects to the fridge's handle by a detachable link, to make manual access possible (e.g., in case of a power outage). We implemented the control on a Revolution Pi IPC [130], utilizing its digital Input-Output (IO) interface, and Ethernet network interface. With that, we combined two RAM layers into one device: the Embedded Controller (EC) and the middleware controller.

We implemented the door opener's SiLA server with the help of the SiLA Python Implementation [131]. We generated a skeleton from the LabwareTransferSiteController, and populated the respective hardware interfaces with the actuator controls. The User Interface (UI) consists of a status light, and a simple control panel with a mode selector, buttons for manual operation, and an emergency stop button.

5.5.5 The liquid handler

After being taken from the fridge, the samples proceed to the sample preparation station. Sample preparation methods (T tasks) might include various pipetting S subtasks, e.g., for various reactions, dilution, pooling, or filtering. For this purpose, we have chosen the Andrew+ pipettor by Waters. Andrew differs from conventional gantry-type liquid handlers by featuring a Selective Compliance Assembly Robot Arm (SCARA) like structure and using conventional hand-held pipettes instead of built-in pumps. This results in a compact and user-friendly design. Combined with an intuitive UI for programming the methods, Andrew effectively fulfills low- to middle-throughput pipetting tasks inautomated laboratories. The OneLab middleware software [132], contrary to conventional pipettor's device-level control software, does not run on a device-dedicated Private Computer (PC) but in the cloud. By default, the vendor's global cloud instance can be utilized, where the user can register their account, and connect the robot via the internet. However, when integration with an over-arching scheduler is desired, a local (on-premises) instance of OneLab Enterprise is required. This can be deployed on a wide range of Linux environments and integrated into the user's intranet, hosting the web-based UI locally. The OneLab backend features a Representational State Transfer (REST)-ful API, which exposes basic functionality, e.g., to start methods and monitor their status. On top of this, API, a SiLA wrapper was created to facilitate standardized interoperability. When the MTP is delivered by the MoMa, the sample preparation T task can be started on Andrew. This is done by the scheduler by calling the appropriate SiLA command.

5.5.6 The MoMa

The active party in the labware transfer in our setup is a MoMa, which implements the *LabwareTransfer**Manipulator**Controller* SiLA feature. The mobERT M5 S unit by EngRoTec [119] is based on the Omron LD-90 AMR [133] and the TM 5-M cobot [134]. It was originally targeted at various intralogistics tasks, e.g., implementing the material flow between a production line's stations. In essence, labware transfer represents the same problem statement, i.e., the pick-and-place handling of objects. However, compared to the basic specification, the design needed to be adapted to the needs of MTP transfer. As such,

Fig. 5.8. The mobERT MoMa by EngRoTec, adapted to labware transportation.

we equipped a Zimmer HRC-03 [135] gripper with a purpose-made finger set specifically designed for MTP gripping. In addition, we incorporated a tool changer unit, to enable the integration of additional end effectors. Furthermore, to make the transportation of multiple MTPs possible at the same time, we equipped the AMR with an on-board storage (hotel). This features ten sites capable of holding MTPs of normal or deep-well height.

The low-level control setup of the MoMa reflects the generic layers of the LAPP RAM. On the *embedded controller* level, it includes the cobot's controller, the AMR's built-in IPC, and a Programmable Logic Controller (PLC) on top. These are on-board components of the MoMa.

Figure 5.8 shows the mobERT mobile manipulator by EngRoTec, deployed in the pharmaceutical sciences laboratory as part of the industrial pilot implementation. In the foreground, the 3D-printed fixture and fiducial markers can be seen, while the robot is performing a pick operation on an MTP. The customized fingertips establish four contact points with the sides of the MTP, ensuring stable grasping. Also visible are the tool changer wrist mount and the integrated wrist camera.

5.5.7 The middleware

The middleware layer also features multiple sub-layers and is deployed on the laboratory-level IPC (located in the control cabinet). Implementing AMR fleet management functionality, Omron's Enterprise Manager [136] runs on a dedicated piece of hardware, which is also installed in the control cabinet. The UI is provided by the Mobile Planner desktop application [137], which we deployed on the IPC. Here runs also the software for the cobot: TM Flow [138]. This software is used for programming and executing the Modular Robot

Program (MRP), which includes the RARs of Q motion sequences, P motion primitives, and A actuator primitives.

These two main middleware pieces are integrated through a superordinate fleet manager software, ERTmiral [139]. Based on the incoming S subtask-level SiLA commands, ERTmiral triggers the respective Q motion sequences on the AMR and cobot. For implementing the SiLA server of the MoMa, we have created a wrapper around ERTmiral's Manufacturing Execution Systems (MES) interface API.

5.5.8 The sequence

For implementing the T `Labware-Transfer` and the underlying sequence, we have utilized the LAPP semantics and hierarchical levels, along with the harmonized SiLA feature definitions.

Figure 5.9 shows a sequence diagram and breakdown of the T `Labware-Transfer` task, which occurs during a laboratory workflow as a supportive activity. The scheduler breaks it down into S subtasks. The sequence starts by sending the S PrepareForOutput command to the MoMa, specifying the following parameters. These are propagated across the subsequent low-level RARs and the corresponding control components.

- **HandoverPositon**: The source site within the source device, located at the source station.
- **InternalPosition**: Site on the on-board hotel to store the MTP during the transort.
- **LabwareType**: Type of the labware, to select the appropriate end effector. (Feature to be implemented.)
- **LabwareUniqueID**: Unique identification number of the piece of labware (e.g., barcode number).

The S PrepareForInput command is broken down by the middleware into Q motion sequences, and the corresponding MRPs are triggered on the AMR and cobot, respectively. These include driving to the tool changer station, swapping the end effector, and driving to the source station.

In the meantime, the S PrepareForOutput command is sent to the fridge door opener, which exposes the handover site. When both preparatory actions are finished, the S GetLabware command is sent to the MoMa. This is broken down to the cobot Q motion sequence for approaching the handover site and grasping the MTP. This Q motion sequence is performed based on the optical marker's frame of reference, as detected by the wrist-mounted camera. The cobot then places the MTP on the designated site of the hotel, and the AMR drives to the temporary position. When the S GetLabware activity is finished, the source device (the fridge) is notified with LabwareRemoved.

The sequence for S PutLabware is analogous. First, the MoMa needs S PrepareForOutput, navigating to the destination station. In the meantime, the destination device is notified to S PrepareForOutput. Then the S PutLabware subtask triggers the corresponding cobot Q motion sequence for placing the MTP at the designated destination site. Finally, the destination device can be notified with LabwareDelivered, and it can begin processing the plate. Also, the MoMa can return to the temporary positon or to the charger, if needed.

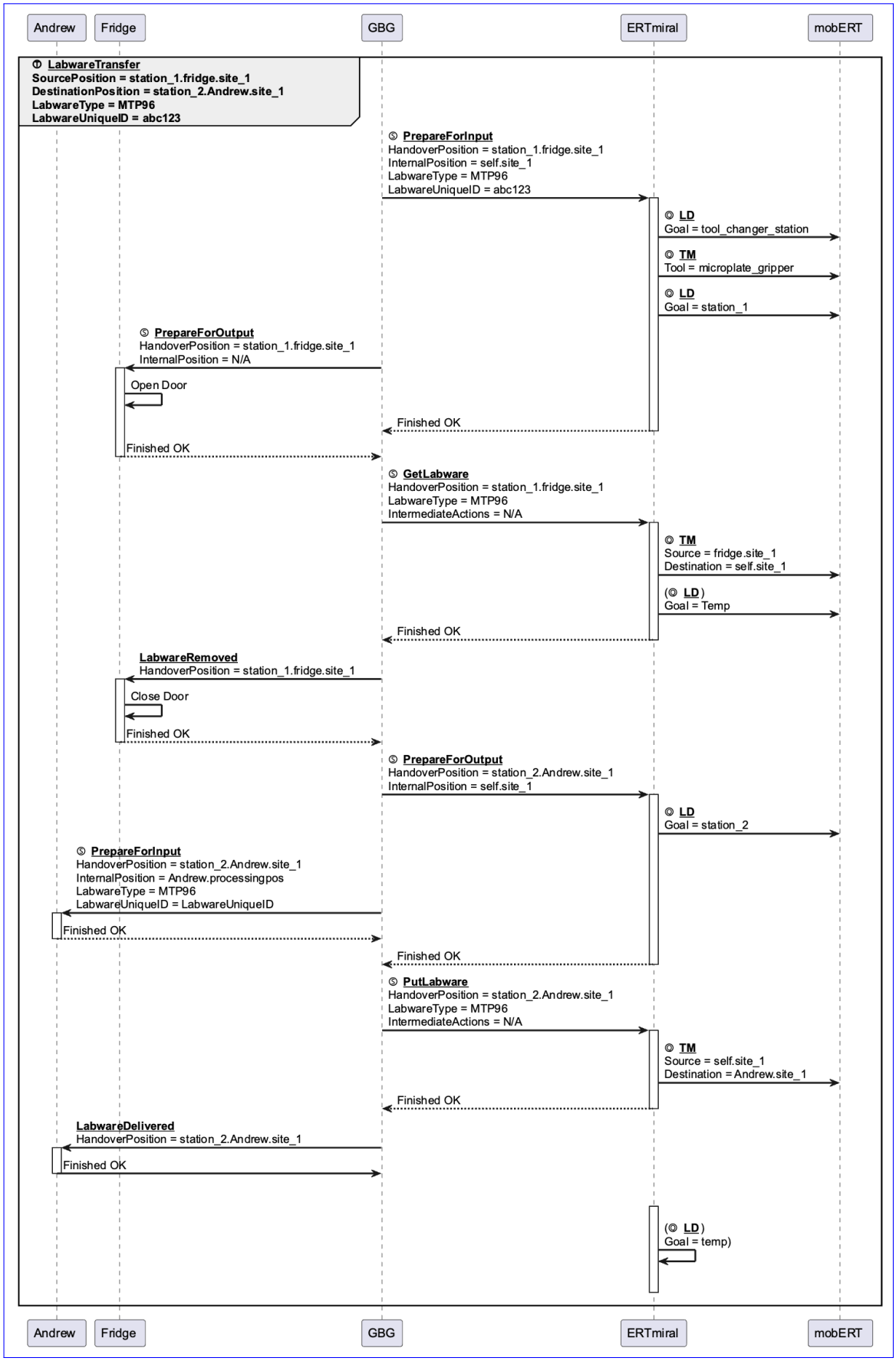


Fig. 5.9. Sequence diagram and breakdown of the `Labware-Transfer` task. Participants are represented as follows: Andrew (pipettor), Fridge (refrigerator and door opener), GBG (scheduler), ERTmiral (middleware), mobERT (MoMa). RARs are underlined, and (T) Tasks, (S) subtasks, and (Q) motion sequences are marked with circled characters, respectively. Parameters and their values are listed below the RAR name. RARs in parentheses are triggered automatically, on demand.

Figure 5.10, analogous to figure 5.6, shows the manifestation of the generic RARs in the form of high-level SiLA commands, and low-level robot-specific elements. In this case, the latter includes the proprietary controls for the LD AMR and the TM cobot. This demonstrates that the high-level RARs are agnostic toward the low-level implementation.

5.6 Testing

We developed and installed the presented system in two stages. First, we deployed the MoMa and the middleware layer in the laboratory. As a first stage, we demonstrated the T `Labware-Transfer` capability without integrating the devices or the scheduler. For this purpose, we placed additively manufactured dummy plate nests, which represented the source and destination sites on two different stations within the laboratory. We programmed a sequence of S subtasks in the middleware, and placed them in a loop to run a continuous stress test. As a result, the MoMa was performing for two consecutive days without an error.

As the second stage, we installed the source and destination devices and the tool changer. Also, we deployed the scheduler software and the SiLA interfaces for both devices and the MoMa system. We demonstrated the execution of the T `Labware-Transfer` task by the means of SiLA commands, first utilizing the Universal SiLA Client [140], then the GBG Scheduler.

5.7 Discussion

Together, the academic prototype and the industrial pilot implementations demonstrate the feasibility of the LAPP concept. These efforts constitute a multi-step endeavor ranging from the scientific conceptualization of the general semantics, information models, and RAM, incorporating these concepts in harmonized industrial standards, to implementing an early-stage prototype and a real-life industrial setup.

Together, the academic prototype and the industrial pilot implementations demonstrate the feasibility of the LAPP concept. These efforts constitute a multi-step endeavor ranging from the scientific conceptualization of the general semantics, information models, and RAM, incorporating these concepts in harmonized industrial standards, to implementing an early-stage prototype and a real-life industrial setup.

Beyond demonstrating successful task execution, the implementations also provide insights directly related to the three theses formulated in this dissertation.

Semantics. The implementations show that the introduction of harmonized semantics and standardized information models is crucial for integration across different sites. In the academic prototype, tasks and subtasks could be described in a machine-readable, vendor-agnostic way using SiLA services and ROS-based abstractions. In the industrial pilot, equivalent principles were applied in a proprietary environment, proving that a common semantic layer allows knowledge and workflows to be transferred between academic and industrial contexts without redefining the logic. This demonstrates that semantics are

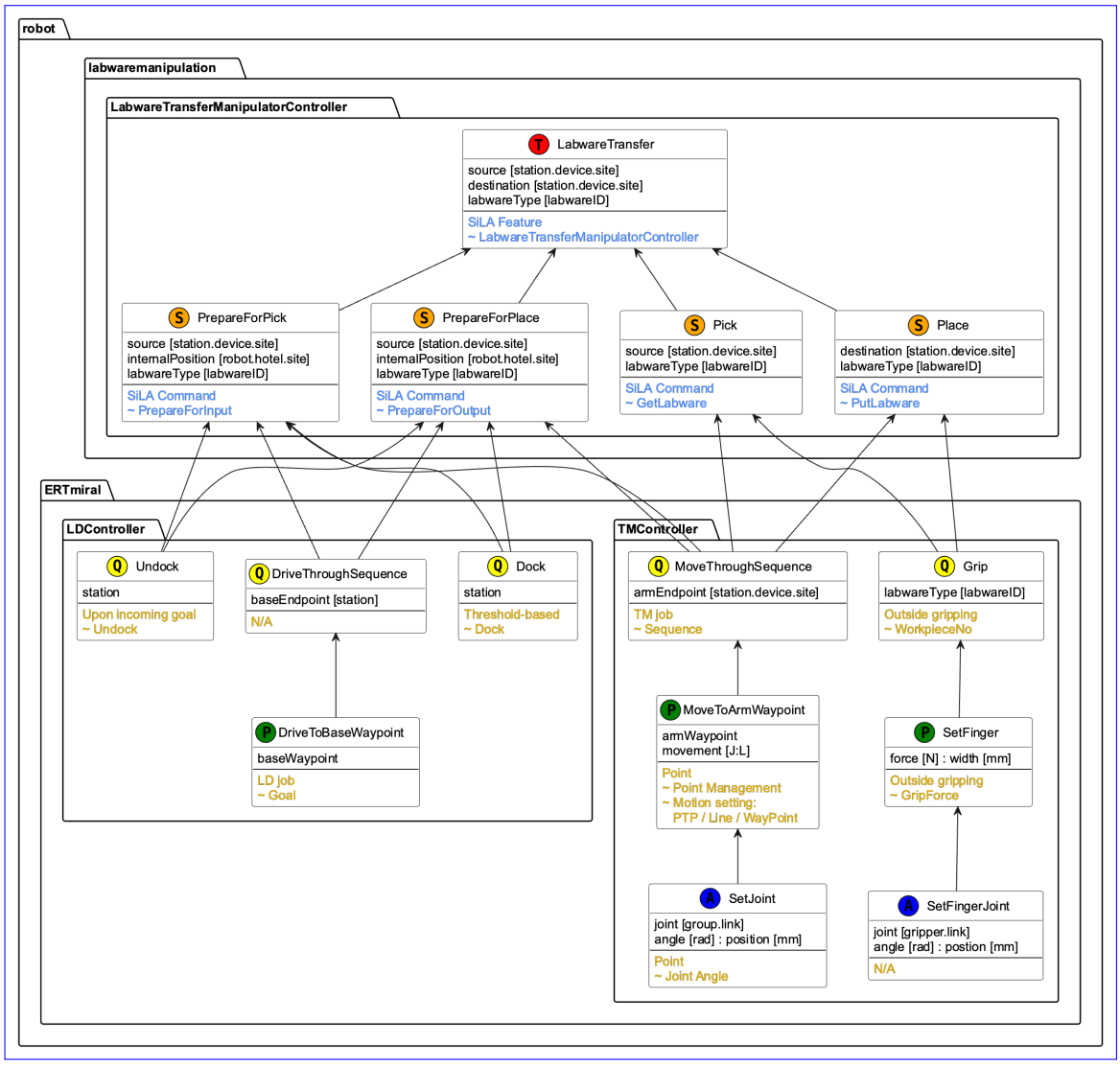


Fig. 5.10. UML-like representation of the labware transfer RARs for the industrial pilot implementation. Namespaces represent a way of categorizing the RARs. The arrows represent how a high-level RAR is composed (broken down into) low-level RARs. Task (T), Subtask (S), Motion sequence (Q), Motion primitive (P) and Actuator primitive (A) are represented as classes. Parameters are shown as class members, with the data type in square brackets []. Overloaded parameters (in the case of polymorphic RARs) are separated by a colon (:). Mapping the SiLA implementation is set in blue font, while the low-level implementation is in yellow. The names of the specific low-level functionalities are marked by a tilde (∼)

not only a conceptual requirement but also an enabler of practical interoperability across heterogeneous environments.

Digital Twin. While the academic prototype prepared to utilize a DT via the ROS `tf` framework, the final implementation relied on manually-thaught marker-based positioning rather than a fully developed, teaching-free DT. This represents an initial step toward digital representation of devices, but not yet a comprehensive digital twin solution. By contrast, the industrial pilot highlighted the necessity of device calibration, qualification, and validation steps that a DT-based approach could formalize and streamline. Together, these results underline both the potential and the current gaps in adopting digital twin practices for laboratory automation. They emphasize that a DT is not optional but will be essential for reducing manual teaching, calibration, and reconfiguration effort, as the concept rolls out.

Reference Architecture Model. The academic and industrial implementations demonstrate the practical applicability of the LAPP RAM. The academic prototype, using open-source technologies, illustrated how a layered approach can be realized in a cost-effective research environment. The industrial pilot, implemented by a commercial integrator, demonstrated that the same layered principles can be applied under the constraints of regulated industry settings. Together, they prove that the LAPP RAM is transferable across sites and contexts, enabling both community-driven development and long-term industrial deployment. This aligns with the arguments made in Section 4.3.4, where violation of the architecture leads to inflexible, non-scalable solutions.

Summary. Taken together, the academic prototype and the industrial pilot not only validated the technical feasibility of the LAPP framework but also provided concrete evidence for the three theses. They showed that harmonized semantics enable interoperability, and that the reference architecture model ensures applicability across diverse environments. The two implementations, though fundamentally different in setting—open-source academic research and proprietary industrial integration—demonstrate the agnostic and transferable nature of the LAPP RAM. They thereby confirm the broader value of the framework as both a technical and organizational blueprint for future laboratory automation systems.

Related publications: [WA6, WA7]

Part 6

SUMMARY

In this thesis, I presented the Laboratory Automation Plug and Play concept. To achieve the goal of harmonizing the integration of supportive robotics in life science laboratories, I first assessed the present-day landscape and collected the preliminaries.

I started chapter 2 with an outline of the capabilities and functionalities of laboratory robots. Based on this assessment, I created a harmonized semantic service description framework centered around the hierarchical decomposition of laboratory workflows. Based on user needs assessment and a targeted survey, I identified the labware transfer activity as a number one activity of interest. This functionality is currently fulfilled mainly by fixed based robots, whereas Mobile Manipulator robots (MoMas) have only recently started to spread across life science laboratory automation applications. I detailed the semantic framework while focusing on the labware transfer activity.

In chapter 3 I presented the information representation pillar of the Laboratory Automation Plug and Play framework, based on the Digital Twin approach. Focusing further on pick-and-place labware transportation, the LAPP DT framework facilitates the offline teaching of supportive laboratory robots and defines their automatic configuration procedure.

As the third pillar, in chapter 4, I spanned an integration system architecture based on principles of industrial automation, including new approaches, such as Service-oriented Architecture (SoA) and Industry 4.0 principles. I drew these mapped to the hierarchical layers of the workflow decomposition framework of chapter 2. Each layer and component of the control ecosystem implements a corresponding level of activity (with an appropriate level of abstraction).

Finally, to demonstrate the feasibility of the three pillars of the LAPP framework, I presented two examples for its implementation. The first was conducted in an academic context as a prototype development effort, using a research-oriented MoMa platform, the TIAGo ++. The basic concepts were implemented using the Standardisation in Laboratory Automation Consortium (SiLA) interoperability protocol. More specifically, the semantic definitions of chapter 2 were implemented as SiLA feature definitions, and the information models of the DT concept (chapter 3) were utilized as harmonized command parameters. The layers and harmonized functionalities of the LAPP Reference Architecture Model (RAM) (chapter 4). The implemented framework comprised the robotic middleware layer based on Robot Operating System (ROS), featuring a SiLA bridge.

The second implementation example was based on an industrial-grade MoMa, called

mobERT, and conducted in a real-life pharmaceutical laboratory setup, constituting a Technology Readiness Level (TRL) 5/6.

These examples together demonstrate the feasibility of the concept.

6.1 Critical Discussion of the Results

The results of this dissertation demonstrated the conceptual feasibility of the LAPP framework and its three pillars—semantic workflow decomposition, digital twin–based information representation, and the reference architecture model—through both academic and industrial pilot implementations. While these implementations provide valuable proof-of-concept, the broader applicability of the framework in real laboratory and industrial contexts requires careful consideration.

A key challenge for the adoption of interoperability frameworks such as SiLA, Laboratory and Analytical Device Standard (LADS), or overarching architectures like LAPP lies in the dynamics of standardization and technology transfer. Frequently, a "chicken-and-egg" problem emerges: solution providers tend to wait until users explicitly demand compliance with specific communication protocols, architectures, or information models, while end users are hesitant to adopt such frameworks until mature, commercially supported solutions are available. This slows down the rollout and wider adoption of best practices, even when their potential benefits are recognized.

Overcoming this impasse requires active dissemination and demonstration. Standardization initiatives must not only define frameworks and protocols, but also showcase their practical utility through tangible examples and pilot implementations. In this regard, the dissertation contributes by aligning the LAPP framework with real-world case studies, thereby providing a bridge between theoretical principles and practical laboratory automation. Nonetheless, the limitations of the current work should be acknowledged. The academic prototype and the industrial pilot address a limited scope—primarily labware transfer—while the full breadth of laboratory workflows and the complexity of multi-laboratory orchestration remain open challenges.

The adoption of the LAPP framework, similar to other interoperability standards, will depend on both technological maturity and community engagement. Broad collaboration between academia, industry, and standardization bodies is essential to ensure that the concepts outlined in this work transition from proof-of-concept to established practice. While the present results represent a first step toward this goal, the author recognizes the difficulties associated with achieving wide-scale adoption of overarching frameworks in laboratory automation and sees dissemination and continued pilot projects as crucial drivers for future progress.

Other publications related to the Ph.D. thesis and the accompanying research work:
[WAO1, WAO2, WAO3, WAO4]

6.2 Future Work

I demonstrated the principal feasibility of the LAPP framework using the SiLA and ROS ecosystems, in terms of implementing the harmonized orchestration of laboratory robots by using the semantic descriptions and incorporating the harmonized system architecture LAPP RAM.

Besides using the harmonized DT parameters in the reference implementations, an asset-centric information framework is yet to be implemented to comprehensively prove the feasibility of the LAPP DT concept.

An endeavor was started within the SiLA Robotics Working Group (SRWG) to implement ontologies for laboratory devices and labware. The discussions revolved around the definition of the terms, where the LAPP DT concept was identified as a suitable starting point to address the needs of robotics and especially to represent the geometrical data, i.e., the positions in the context of the laboratory device and labware. However, in other ecosystems besides SiLA, such frameworks for hierarchical position representation already exist, providing a technological basis for the implementation of the LAPP DT. As such, the ROS tf framework [141] constitutes a versatile solution for managing coordinate transformations within a robotic ecosystem. Combining this with the appropriate persistence solution, i.e., a DT prototype representation represents a suitable next step.

Furthermore, in a MoveIt planning scene [142], labware can be represented as manipulable objects, and the corresponding grasping strategies can be modeled. Also, devices can be represented through their collision geometry and positions as goals for motion sequences. Populating these from a persistent DT instance representation from the cloud and using the ROS parameter server for DT instance parameters constitutes a desirable way forward to implement the LAPP DT.

The SiLA ontology framework could serve as the persistence DT parameter backbone. Queried e.g., via SiLA interface, such a database could fulfill the LAPP DT functionality.

Other ecosystems besides SiLA, especially those available in industrial and process automation context, based on Open Platform Communications - Unified Architecture (OPC-UA) are worth considering. These include off-the-shelf solutions, such as the Relative Spatial Locations (RSL) framework for representing hierarchical position data, similar to the tf framework of ROS.

Being based on OPC-UA the LADS protocol constitutes an attractive alternative to SiLA. Creating reference implementations for the LAPP concept, especially for the DT framework, based on LADS and OPC-UA can further prove the versatility and feasibility of the LAPP concept, thus it is named as a desirable future work package, as a continuation of this present thesis work.

Furthermore, the implementation of additional robotic capabilities in laboratory automation, besides the pick-and-place type labware transfer can be based on the taxonomy presented in chapter 2 and populate the outlined structure. This constitutes the continuation of ongoing early-stage research I cited in the corresponding taxonomy. These include, on the high-level, advanced cognition and perception based solutions, where the middle-to-low level controls of the robotic activities (from motion sequence down) are not hard-coded (pre-programmed relative positions), but defined by the means of abstract semantic T task and S subtask representations. It is the semantic and geometric planner's

role to calculate and execute the robotic Q motion sequences, P motion primitives and A actuator primitives (i.e., the corresponding trajectories), as Leidner proposed [79].

These further developments will enable more complex workflow implementations across multi-laboratory deployments, where one or multiple robots in a fleet perform a wide range of applications beyond sample transportation. Achieving this requires an expansion of the LAPP-RAM, with particular emphasis on over-arching scalability. In addition to the middleware-layer components, greater focus will need to be placed on higher-level orchestration, including schedulers and overarching coordination layers that can connect multiple laboratories and dispatch heterogeneous robotic systems to different use cases across sites.

New advances in cognitive robotics and semantic control of high-level tasks will further contribute to this perspective, supported by AI-based solutions that are currently being developed in the context of humanoid robots. For cross-laboratory applications, however, wheeled mobile robots are likely to remain the more practical option compared to legged or humanoid platforms. The Robotic Activity Representations (RARs)—such as “MoveThroughSequence” for manipulators and “DriveThroughSequence” for mobile bases—were designed specifically to decouple high-level task logic from the low-level details of locomotion and control. This abstraction also extends naturally to legged platforms, where a quadruped or humanoid can be treated as a mobile base executing the same high-level RAR, while its internal gait generation and foothold planning remain implementation-specific. In such cases, motion sequences and actuator primitives can be masked behind the high-level control components and implemented by different techniques, including both model-based and machine learning–based approaches.

In our internal assessments for pharmaceutical process-development laboratories, quadruped robots appeared most promising as inspection and error-handling agents, rather than as high-precision sample transfer systems [WAO3]. Commercially supported capabilities of platforms such as the Spot Arm emphasize door and valve actuation and basic object manipulation, while integrated payloads such as the Spot CAM+IR are well suited to remote situational awareness and first-response tasks. These strengths contrast with the tight-tolerance pick-and-place operations required for laboratory sample handling, where wheeled mobile robots and stationary manipulators remain the preferred choice.

Overall the LAPP framework’s versatility and modularity make it suitable to cover both traditional and modern robot control approaches:

- The most basic method is manual teaching, by manually moving and jogging the arm and storing the positions in the context of the robot.
- Offline teaching utilizing the DT approach and defining the actions in the context of each laboratory device, eliminating the need for manual teaching.
- The most advanced technique is incorporating advanced service robotics technologies, where, considering the complexity of the environment and the tasks, the activities are not hard-coded, but, as described above, calculated by advanced robotic cognition techniques.

REFERENCES

[1] B. Lightsey, *Systems Engineering Fundamentals*. Defense Acquisition University Press, 2001.

[2] S. A. Brotherton, R. T. Friend, and E. S. Norman, "Applying the work breakdown structure to the project management lifecycle," in *PMI global congress proceedings*, Denver, CO., 2008, pp. 1–15. [Online]. Available: https://www.pmi.org/learning/library/applying-work-breakdown-structure-project-lifecycle-6979

[3] "Difference and use cases of Jira issue types: Epic... - Atlassian Community." [Online]. Available: https://community.atlassian.com/t5/Jira-articles/Difference-and-use-cases-of-Jira-issue-types-Epic-vs-Story-vs/ba-p/1655157

[4] "Epics, Stories, Themes, and Initiatives | Atlassian." [Online]. Available: https://www.atlassian.com/agile/project-management/epics-stories-themes

[5] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the Digital Twin: A systematic literature review," *CIRP Journal of Manufacturing Science and Technology*, vol. 29, pp. 36–52, 2020. [Online]. Available: https://doi.org/10.1016/j.cirpj.2020.02.002

[6] B. Tipary, "Assembly plan calibration using sensor networks," Ph.D. dissertation, 2022.

[7] G. Samorini, "The oldest archeological data evidencing the relationship of Homo sapiens with psychoactive plants: A worldwide overview," *Journal of Psychedelic Studies*, vol. 3, no. 2, pp. 63–80, 6 2019. [Online]. Available: https://akjournals.com/view/journals/2054/3/2/article-p63.xml

[8] "Pharmacy | Drug Compounding & Dispensing, Pharmaceutical Science | Britannica." [Online]. Available: https://www.britannica.com/science/pharmacy

[9] I. Masic, "Contribution of Arabic Medicine and Pharmacy to the Development of Health Care Protection in Bosnia and Herzegovina - the Second Part," *Medical archives (Sarajevo, Bosnia and Herzegovina)*, vol. 71, no. 6, pp. 439–448, 12 2017.

[10] J. S. Crawshaw, "Families, medical secrets and public health in early modern Venice," *Renaissance Studies*, vol. 28, no. 4, pp. 597–618, 9 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1111/rest.12081https://onlinelibrary.wiley.com/doi/abs/10.1111/rest.12081https://onlinelibrary.wiley.com/doi/10.1111/rest.12081

[11] A. W. Jones, "Early drug discovery and the rise of pharmaceutical chemistry," *Drug Testing and Analysis*, vol. 3, no. 6, pp. 337–344, 6 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/dta.301https://onlinelibrary.wiley.com/doi/abs/10.1002/dta.301https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/dta.301

[12] K. Olsen, "The first 110 years of laboratory automation: Technologies, applications, and the creative scientist," *Journal of Laboratory Automation*, vol. 17, no. 6, pp. 469–480, 12 2012.

[13] V. Gupta, M. Sengupta, J. Prakash, and B. C. Tripathy, "An Introduction to Biotechnology," *Basic and Applied Aspects of Biotechnology*, pp. 1–21, 2017. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-10-0875-7_1

[14] J. P. Hughes, S. S. Rees, S. B. Kalindjian, and K. L. Philpott, "Principles of early drug discovery," *British Journal of Pharmacology*, vol. 162, no. 6, p. 1239, 3 2011. [Online]. Available: /pmc/articles/PMC3058157//pmc/articles/PMC3058157/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC3058157/

[15] J. W. Scannell, A. Blanckley, H. Boldon, and B. Warrington, "Diagnosing the decline in pharmaceutical R&D efficiency," *Nature Reviews Drug Discovery*, vol. 11, no. 3, pp. 191–200, 2012.

[16] J. Hall, S. Matos, S. Gold, and L. S. Severino, "The paradox of sustainable innovation: The 'Eroom' effect (Moore's law backwards)," *Journal of Cleaner Production*, vol. 172, pp. 3487–3497, 1 2018.

[17] A. Mobley, S. K. Linder, R. Braeuer, L. M. Ellis, and L. Zwelling, "A Survey on Data Reproducibility in Cancer Research Provides Insights into Our Limited Ability to Translate Findings from the Laboratory to the Clinic," *PLoS ONE*, vol. 8, no. 5, 5 2013.

[18] M. Baker, "1,500 scientists lift the lid on reproducibility," *Nature*, vol. 533, no. 7604, pp. 452–454, 2016.

[19] J. P. Ioannidis, "Why Most Clinical Research Is Not Useful," *PLoS Medicine*, vol. 13, no. 6, 6 2016.

[20] "Lab automation - Goldfuß engineering GmbH." [Online]. Available: https://www.goldfuss-engineering.com/en/lab-automation.html

[21] "Concept | Robotic Biology Institute Inc. | RBI." [Online]. Available: https://rbi.co.jp/en/concept/

[22] S. Konstantinidis, H.-Y. Goh, J. M. Martin Bufájer, P. de Galbert, M. Parau, and A. Velayudhan, "Flexible and Accessible Automated Operation of Miniature Chromatography Columns on a Liquid Handling Station," *Biotechnology Journal*, vol. 13, no. 3, p. 1700390, 3 2018.

[23] M. Altekar, C. A. Homon, M. A. Kashem, S. W. Mason, R. M. Nelson, L. A. Patnaude, J. Yingling, and P. B. Taylor, "Assay Optimization: A Statistical Design of Experiments Approach," *Journal of Laboratory Automation*, vol. 11, no. 1, pp. 33–41, 2006.

[24] C. Rumford Walker, *Toward the Automatic Factory: A Case Study of Men and Machines*. Praeger, 1977.

[25] I. Burckhardt, "Laboratory Automation in Clinical Microbiology," *Bioengineering*, vol. 5, no. 4, p. 102, 3 2018.

[26] A. Croxatto, G. Prod'hom, F. Faverjon, Y. Rochais, and G. Greub, "Laboratory automation in clinical bacteriology: What system to choose?" *Clinical Microbiology and Infection*, vol. 22, no. 3, pp. 217–235, 3 2016.

[27] "Andrew Alliance - Pipetting tools and software for the scientific lab." [Online]. Available: https://www.andrewalliance.com/

[28] "Biosero Acceleration Lab," [cited 2021 Oct 18]. [Online]. Available: https://biosero.com/integrations/acceleration-lab/

[29] S. Kleine-Wechelmann, K. Bastiaanse, M. Freundel, and C. Becker-Asano, "Designing the mobile robot Kevin for a life science laboratory," in *RO-MAN 2022 - 31st IEEE International Conference on Robot and Human Interactive Communication: Social, Asocial, and Antisocial Robots*. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 870–875.

[30] N. Yoshikawa, M. Skreta, K. Darvish, S. Arellano-Rubach, Z. Ji, L. Bjørn Kristensen, A. Z. Li, Y. Zhao, H. Xu, A. Kuramshin, A. Aspuru-Guzik, F. Shkurti, and A. Garg, "Large language models for chemistry robotics," *Autonomous Robots*, 2023. [Online]. Available: https://doi.org/10.1007/s10514-023-10136-2

[31] C. D. Hawker, "Laboratory Automation: Total and Subtotal," *Clinics in Laboratory Medicine*, vol. 27, no. 4, pp. 749–770, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0272271207000856

[32] G. Lippi and G. D. Rin, "Advantages and limitations of total laboratory automation: a personal
} overview," *Clinical Chemistry and Laboratory Medicine (CCLM)*, vol. 57, no. 6, pp. 802–811, 2019. [Online]. Available: https://www.degruyter.com/document/doi/10.1515/cclm-2018-1323/html?lang=enhttp://dx.doi.org/10.1515/cclm-2018-1323

[33] "J3016_202104: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles - SAE International," 2021. [Online]. Available: https://www.sae.org/standards/content/j3016_202104/

[34] T. D. Nagy and T. Haidegger, "Performance and Capability Assessment in Surgical Subtask Automation," *Sensors*, vol. 22, no. 7, p. 2501, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/7/2501

[35] J. Beal and M. Rogers, "Levels of autonomy in synthetic biology engineering," *Molecular Systems Biology*, vol. 16, no. 12, p. e10019, 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7744957/

[36] H. Fleischer and K. Thurow, *Automation Solutions for Analytical Measurements: Concepts and Applications*. John Wiley & Sons, 2017.

[37] P. Courtney, "New trends in intelligent robotics in the laboratory," *European Pharmaceutical Review*, vol. 21, no. 2, pp. 36–38, 2016. [Online]. Available: www.euroc-project.eu

[38] P. Groth and J. Cox, "Indicators for the use of robotic labs in basic biomedical research: A literature analysis," *PeerJ*, vol. 2017, no. 11, p. e3997, 11 2017. [Online]. Available: https://peerj.com/articles/3997

[39] H. Fleischer, D. Baumann, X. Chu, T. Roddelkopf, M. Klos, and K. Thurow, "Integration of Electronic Pipettes into a Dual-arm Robotic System for Automated Analytical Measurement Processes Behaviors," *IEEE International Conference on Automation Science and Engineering*, vol. 2018-Augus, pp. 22–27, 2018.

[40] X. Chu, H. Fleischer, T. Roddelkopf, N. Stoll, M. Klos, and K. Thurow, "A LC-MS integration approach in life science automation: Hardware integration and software integration," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 3 2015, pp. 979–984.

[41] H. Fleischer, D. Baumann, S. Joshi, X. Chu, T. Roddelkopf, M. Klos, and K. Thurow, "Analytical measurements and efficient process generation using a dual–arm robot equipped with electronic pipettes," *Energies*, vol. 11, no. 10, p. 2567, 10 2018.

[42] "Robots and robotic devices — Collaborative robots, ISO/TS 15066:2016," 2016.

[43] J. Bohren, R. B. Rusu, E. G. Jones, E. Marder-Eppstein, C. Pantofaru, M. Wise, L. Mösenlechner, W. Meeussen, and S. Holzer, "Towards autonomous robotic butlers: Lessons learned with the PR2," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 5568–5575.

[44] M. Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich, "Fetch & Freight: Standard Platforms for Service Robot Applications," *Fetch Robotics Inc.*, 2018.

[45] J. Pages, L. Marchionni, and F. Ferro, "TIAGo: the modular robot that adapts to different research needs," *PAL Robotics S.L.*, p. 4, 2016.

[46] A. Abduljalil, "An intelligent multi-floor mobile robot transportation system in life science laboratories," Ph.D. dissertation, University of Rostock, 2019.

[47] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 6 2014.

[48] "Precise Automation - PreciseFlex 400 Sample Handler," [cited 2021 May 19]. [Online]. Available: http://preciseautomation.com/SampleHandler.html

[49] T. Brode and A. Traube, "Mobile Lab Robot (Kevin) - Laboratory automation thought differently," [cited 2021 May 19]., 2019. [Online]. Available: https://www.messe.de/apollo/labvolution_2019/obs/Binary/A921771/21539_38579_155489679430865800_aeid_205__265_atomfeld_205_265-1314901554897996.pdf_org

[50] "UniteLabs AG – Lab automation software company," [cited 2021 May 17]. [Online]. Available: https://unitelabs.ch/

[51] "KMR iiwa | KUKA AG," [cited 2021 May 17]. [Online]. Available: https://www.kuka.com/en-hu/products/mobility/mobile-robot-systems/kmr-iiwa

[52] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, N. Rankin, B. Harris, R. S. Sprick, and A. I. Cooper, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 7 2020.

[53] "Autonomous robotics in pharma manufacturing," [cited 2021 Apr 22]. [Online]. Available: https://www.pmgroup-global.com/news/pharmaceutical-robotics/

[54] A. K. Ramasubramanian and N. Papakostas, "Operator - Mobile robot collaboration for synchronized part movement," in *Procedia CIRP*, vol. 97. Elsevier B.V., 2020, pp. 217–223.

[55] "SiLA Rapid Integration," 2018. [Online]. Available: https://sila-standard.com/

[56] "SiLA 2 Part (A) - Overview, Concepts and Core Specification." [Online]. Available: https://docs.google.com/document/d/1nGGEwbx45ZpKeKYH18VnNysREbr1EXH6FqlCo03yASM/edit?usp=embed_facebook

[57] H. Bär, R. Hochstrasser, and B. Papenfuß, "SiLA: Basic Standards for Rapid Integration in Laboratory Automation," *Journal of Laboratory Automation*, vol. 17, no. 2, pp. 86–95, 2012. [Online]. Available: https://doi.org/10.1177/2211068211424550

[58] "Lab Automation Solutions 25+ years Astech Projects." [Online]. Available: https://astechprojects.co.uk/expertise/laboratory-automation

[59] A. Brendel, F. Dorfmüller, A. Liebscher, P. Kraus, K. Kress, H. Oehme, M. Arnold, and R. Koschitzki, "Laboratory and Analytical Device Standard (LADS): A Communication Standard Based on OPC UA for Networked Laboratories," *Advances in Biochemical Engineering/Biotechnology*, vol. 182, pp. 175–194, 2022. [Online]. Available: https://link.springer.com/chapter/10.1007/10_2022_209

[60] "Laboratory and Analytical Device Standard - OPC Foundation." [Online]. Available: https://opcfoundation.org/developer-tools/documents/view/208

[61] "Technology Readiness Level - an overview ScienceDirect Topics." [Online]. Available: https://www.sciencedirect.com/topics/engineering/technology-readiness-level

[62] "Momentum™ Workflow Scheduling Software," [cited 2021 Apr 10]. [Online]. Available: https://www.thermofisher.com/order/catalog/product/MOMENTUM#/MOMENTUM

[63] B. Coulter, "SAMI EX software, Biomek, Automation - Beckman Coulter." [Online]. Available: https://www.beckman.com/liquid-handlers/software/sami-ex

[64] "Lucullus – Securecell website," [cited 2021 Apr 10]. [Online]. Available: https://securecell.ch/en/lucullus/

[65] "Green Button Go Scheduler | Biosero." [Online]. Available: https://biosero.com/software/green-button-go-scheduler/

[66] "PlateButler® Software V3.6 | Lab Services," [cited 2021 Apr 10]. [Online]. Available: https://www.lab-services.nl/en/products/platebutler/platebutler-software-v36

[67] "Overlord Scheduling Software - Peak Analysis & Automation," [cited 2021 Apr 10]. [Online]. Available: https://paa-automation.com/products/overlord-scheduling-software/

[68] "Softlinx Lab Automation Software - Interface, Scheduling & Automation," [cited 2021 Apr 10]. [Online]. Available: https://hudsonrobotics.com/lab-automation-software/softlinx/

[69] "UniteFlow – UniteLabs AG," [cited 2021 Apr 10]. [Online]. Available: https://unitelabs.ch/technology/uniteflow/

[70] "Dynamic Scheduler," [cited 2021 Apr 10]. [Online]. Available: https://www.equicon.de/en/laboratory-automation/scheduling-software-nicelab/scheduler

[71] "LARAsuite / LARA · GitLab." [Online]. Available: https://gitlab.com/LARAsuite/lara

[72] M. Dörr and U. T. Bornscheuer, "Program-guided design of high-throughput enzyme screening experiments and automated data analysis/evaluation," *Methods in Molecular Biology*, vol. 1685, pp. 269–282, 2018. [Online]. Available: https://link.springer.com/protocol/10.1007/978-1-4939-7366-8_16

[73] B. Bartley, J. Beal, M. Rogers, R. B. B. N. Technologies, D. Bryce, and R. P. Goldman, "Building an Open Representation for Biological Protocols," *[Preprint]*, vol. 1, no. 1, 2022. [Online]. Available: https://github.com/Bioprotocols/PAML-specification

[74] "XDL 2.0 Standard — xdl 2.0.1.dev12+g8fdf586b documentation." [Online]. Available: https://croningroup.gitlab.io/chemputer/xdl/standard/index.html

[75] L. Cronin, S. Pagel, and A. Sharma, "Chemputer and Chemputation – A Universal Chemical Compound Synthesis Machine," 8 2024. [Online]. Available: https://arxiv.org/pdf/2408.09171

[76] X. Chu, "Automation Strategies for Sample Preparation in Life Science Applications," Ph.D. dissertation, 2016.

[77] T. D. Nagy and T. Haidegger, "A DVRK-based Framework for Surgical Subtask Automation," *Acta Polytechnica Hungarica*, vol. 16, no. 8, 2019.

[78] S. S. Vedula, A. O. Malpani, L. Tao, G. Chen, Y. Gao, P. Poddar, N. Ahmidi, C. Paxton, R. Vidal, S. Khudanpur, G. D. Hager, and C. C. G. Chen, "Analysis of the structure of surgical activity for a suturing and knot-tying task," *PLoS ONE*, vol. 11, no. 3, 3 2016.

[79] D. Leidner, C. Borst, and G. Hirzinger, "Things are made for what they are: Solving manipulation tasks by using functional object classes," *IEEE-RAS International Conference on Humanoid Robots*, pp. 429–435, 2012.

[80] "Universal Robots e-Series User Manual UR10e Original instructions (en) UR10e User Manual," 2009.

[81] D. Knobbe, H. Zwirnmann, M. Eckhoff, and S. Haddadin, "Core Processes in Intelligent Robotic Lab Assistant: Flexible Liquid Handling," *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2335–2342, 2022.

[82] D. S. Leidner, "Cognitive Reasoning for Compliant Robot Manipulation," *Doctoral Thesis*, p. 211, 2017. [Online]. Available: http://www.springer.com/series/5208

[83] H. Fleischer, R. R. Drews, J. Janson, B. R. Chinna Patlolla, X. Chu, M. Klos, and K. Thurow, "Application of a Dual-Arm Robot in Complex Sample Preparation and Measurement Processes," *Journal of Laboratory Automation*, vol. 21, no. 5, pp. 671–681, 10 2016.

[84] "feature_definitions/org/silastandard/instruments/labware/manipulation · master · SiLA2 / sila_base · GitLab." [Online]. Available: https://gitlab.com/SiLA2/sila_base/-/tree/master/feature_definitions/org/silastandard/instruments/labware/manipulation

[85] "feature_definitions/hu/uniobuda/irob · unified_robot_features · \'{A}dám Wolf / sila_base_robotics · GitLab," 2022. [Online]. Available: https://gitlab.com/adam.wolf1/sila-base-robotics/-/tree/unified_robot_features/feature_definitions/hu/uniobuda/irob

[86] M. Grieves and J. Vickers, "Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems," *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*, no. August, pp. 85–113, 2016.

[87] R. Rosen, G. Von Wichert, G. Lo, and K. D. Bettenhausen, "About the importance of autonomy and digital twins for the future of manufacturing," *IFAC-PapersOnLine*, vol. 28, no. 3, pp. 567–572, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.ifacol.2015.06.141

[88] E. Negri, L. Fumagalli, and M. Macchi, "A Review of the Roles of Digital Twin in CPS-based Production Systems," *Procedia Manufacturing*, vol. 11, pp. 939–948, 2017. [Online]. Available: www.sciencedirect.com

[89] Y. Lu, C. Liu, K. I. Wang, H. Huang, and X. Xu, "Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robotics and Computer-Integrated Manufacturing*, vol. 61, 2 2020. [Online]. Available: https://doi.org/10.1016/j.rcim.2019.101837

[90] B. Schleich, N. Anwer, L. Mathieu, and S. Wartzack, "Shaping the digital twin for design and production engineering," *CIRP Annals - Manufacturing Technology*, vol. 66, no. 1, pp. 141–144, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.cirp.2017.04.040

[91] I. Verner, D. Cuperman, A. Fang, M. Reitman, T. Romm, and G. Balikin, "Robot Online Learning Through Digital Twin Experiments: A Weightlifting Project," *Lecture Notes in Networks and Systems*, vol. 22, pp. 307–314, 2018. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-64352-6_29

[92] C. E. Magrin, G. Del Conte, and E. Todt, "Creating a Digital Twin as an Open Source Learning Tool for Mobile Robotics," *2021 Latin American Robotics Symposium, 2021 Brazilian Symposium on Robotics, and 2021 Workshop on Robotics in Education, LARS-SBR-WRE 2021*, pp. 13–18, 2021.

[93] A. A. Malik and A. Bilberg, "Digital twins of human robot collaboration in a production setting," *Procedia Manufacturing*, vol. 17, pp. 278–285, 1 2018.

[94] E. Pairet, P. Ardón, X. Liu, J. Lopes, H. Hastie, and K. S. Lohan, "A Digital Twin for Human-Robot Interaction," *ACM/IEEE International Conference on Human-Robot Interaction*, vol. 2019-March, p. 372, 3 2019.

[95] P. Baranyi, A. Csapó, T. Budai, and G. Wersényi, "Introducing the Concept of Internet of Digital Reality-Part I," *Acta Polytechnica Hungarica*, vol. 18, no. 7, p. 2021.

[96] M. D. Vu, T. N. Nguyen, and C. A. My, "Design and Implementation of a Digital Twin to Control the Industrial Robot Mitsubishi RV-12SD," *Mechanisms and Machine Science*, vol. 113 MMS, pp. 422–432, 12 2021. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-91892-7_40

[97] V. Kuts, M. Sarkans, T. Otto, T. Tähemaa, and Y. Bondarenko, "Digital Twin: Concept of Hybrid Programming for Industrial Robots — Use Case," *ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE)*, vol. 2B-2019, 1 2020.

[98] G. Erdős, I. Paniti, and B. Tipary, "Transformation of robotic workcells to digital twins," *CIRP Annals*, vol. 69, no. 1, pp. 149–152, 1 2020.

[99] B. Tipary and G. Erdős, "Generic development methodology for flexible robotic pick-and-place workcells based on Digital Twin," *Robotics and Computer-Integrated Manufacturing*, vol. 71, p. 102140, 10 2021.

[100] A. Sagitov, K. Shabalina, L. Sabirova, H. Li, and E. Magid, "ARTag, AprilTag and CALTag fiducial marker systems: Comparison in a presence of partial marker occlusion and rotation," in *ICINCO 2017 - Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics*, vol. 2. SciTePress, 2017, pp. 182–191.

[101] F. Bergamasco, A. Albarelli, and A. Torsello, "Pi-Tag: A fast image-space marker design based on projective invariants," *Machine Vision and Applications*, vol. 24, no. 6, pp. 1295–1310, 2013.

[102] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics (intelligent robotics and autonomous agents series)*, 2005, vol. 45. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Probabilistic+Robotics+(Intelligent+Robotics+and+Autonomous+Agents)#0

[103] "feature_definitions/ch/unitelabs · master · SiLA2 / sila_base · GitLab." [Online]. Available: https://gitlab.com/SiLA2/sila_base/-/tree/master/feature_definitions/ch/unitelabs

[104] "(224) Webinar Kevin - YouTube." [Online]. Available: https://www.youtube.com/watch?v=HfZCNi841Oo&t=27s

[105] "Unified Architecture - OPC Foundation," [cited 2021 Apr 19]. [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-ua/

[106] "Markets & Collaboration - OPC Foundation," [cited 2021 May 13]. [Online]. Available: https://opcfoundation.org/markets-collaboration/

[107] "Reference Architecture Model Industrie 4.0 (RAMI4.0) English translation of DIN SPEC 91345:2016-04," pp. 1–40, 2016.

[108] "Plattform Industrie 4.0 - Asset Administration Shell Specifications," [cited 2021 Oct 18]. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Standardartikel/specification-administrationshell.html

[109] "BPMN Specification - Business Process Model and Notation." [Online]. Available: https://www.bpmn.org/

[110] "ISA 95.00.01-2010 Enterprise-Control System Integration - Part 1: Models and Terminology."

[111] H. Lasi, P. Fettke, H. G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business and Information Systems Engineering*, vol. 6, no. 4, pp. 239–242, 8 2014. [Online]. Available: https://link.springer.com/article/10.1007/s12599-014-0334-4

[112] F. Schnicke, T. Kuhn, and P. O. Antonino, "Enabling industry 4.0 service-oriented architecture through digital twins," *Communications in Computer and Information Science*, vol. 1269 CCIS, pp. 490–503, 2020. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-59155-7_35

[113] "Standards SiLA Rapid Integration," 2017. [Online]. Available: https://sila-standard.com/standards/

[114] H. Fakhruldeen, G. Pizzuto, J. Glowacki, and A. I. Cooper, "ARChemist: Autonomous Robotic Chemistry System Architecture," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 6013–6019, 2022.

[115] P. Shiri, V. Lai, T. Zepel, D. Griffin, J. Reifman, S. Clark, S. Grunert, L. P. Yunker, S. Steiner, H. Situ, F. Yang, P. L. Prieto, and J. E. Hein, "Automated solubility screening platform using computer vision," *iScience*, vol. 24, no. 3, p. 102176, 3 2021.

[116] "PerceptiveAPC - PharmaMV." [Online]. Available: https://www.perceptiveapc.com/software/pharmamv/

[117] "ECL Documentation." [Online]. Available: https://www.emeraldcloudlab.com/documentation/functions/

[118] "How Cloud Laboratories Work." [Online]. Available: https://www.emeraldcloudlab.com/how-it-works/

[119] "mobERT® - EngRoTec - Solutions GmbH." [Online]. Available: https://www.engrotec-solutions.de/en/mobert/

[120] "BioLAGO." [Online]. Available: https://www.biolago.org/en/

[121] "SiLA2 / sila_ros · GitLab." [Online]. Available: https://gitlab.com/SiLA2/sila_ros

[122] A. Wolf, "BioSASH-4 Intro," 2022. [Online]. Available: https://docs.google.com/presentation/d/18KRi_EXG3U2Xs1hK5KAfGyVQQf6D9hRW

[123] S. Koch, "SiLA2 Labware Transfer Controller Features," 2023. [Online]. Available: https://docs.google.com/presentation/d/1yxWksPMuKrcYjP_ZYytU_k0v7LIVXyQh

[124] "SiLA2 / sila_robotics / sila_pick_and_place_example · GitLab," 2022. [Online]. Available: https://gitlab.com/SiLA2/sila_robotics/sila_pick_and_place_example

[125] A. Wolf, "BioSASH-4 Results," 2022. [Online]. Available: https://docs.google.com/presentation/d/1GKa13sUpJGoAOyef5O5UbaAh5CQGgFOd

[126] "sila_ros_bridge · master · SiLA2 / sila_robotics / sila_ros · GitLab," 2022. [Online]. Available: https://gitlab.com/SiLA2/sila_robotics/sila_ros/-/tree/master/sila_ros_bridge

[127] "Robots/TIAGo/Tutorials/motions/play_motion - ROS Wiki," 2023. [Online]. Available: http://wiki.ros.org/Robots/TIAGo/Tutorials/motions/play_motion

[128] S. R. Sadin, F. P. Povinelli, and R. Rosen, "The NASA technology push towards future space mission systems," in *Acta Astronautica*, vol. 20, 1988, pp. 73–77. [Online]. Available: https://ntrs.nasa.gov/citations/19890030268

[129] "feature_definitions/org/silastandard/instruments/labware/manipulation · master · SiLA2 / sila_base · GitLab," 2023. [Online]. Available: https://gitlab.com/SiLA2/sila_base/-/tree/master/feature_definitions/org/silastandard/instruments/labware/manipulation

[130] "Meet the Revolution Pi products - Industrial Raspberry Pi." [Online]. Available: https://revolutionpi.com/en/revolution-pi-series

[131] "SiLA2 / sila_python · GitLab," 2023. [Online]. Available: https://gitlab.com/SiLA2/sila_python

[132] "Design & Execute Laboratory Protocols | OneLab | Andrew Alliance." [Online]. Available: https://www.andrewalliance.com/laboratory-software/

[133] "LD-series | OMRON, Europe." [Online]. Available: https://industrial.omron.eu/en/products/ld-series

[134] "Techman Robot | TM5 - 900." [Online]. Available: https://www.tm-robot.com/en/tm5-900/

[135] "HRC-03 - Zimmer Group." [Online]. Available: https://www.zimmer-group.com/en/technologies-components/components/handling-technology/grippers/hrc/collaborative/2-jaw-parallel-grippers/hrc-03

[136] "FLOW Core Fleet Manager | OMRON, Europe." [Online]. Available: https://industrial.omron.eu/en/products/flow-core-fleet-manager

[137] "OMRON AMR technology and software explained | OMRON, Europe." [Online]. Available: https://industrial.omron.eu/en/products/amr-technology--software/autonomous-mobile-robot-technology-and-software-explained

[138] "TMflow | Techman Robot." [Online]. Available: https://www.tm-robot.com/en/tmflow/

[139] "ERTmiral - EngRoTec - Solutions GmbH." [Online]. Available: https://www.engrotec-solutions.de/ertmiral-flottenmanager/

[140] "Files · master · SiLA2 / universal_sila_client / sila_universal_client · GitLab." [Online]. Available: https://gitlab.com/SiLA2/universal-sila-client/sila_universal_client/-/tree/master

[141] "tf - ROS Wiki." [Online]. Available: http://wiki.ros.org/tf

[142] "moveit_core: Planning Scene." [Online]. Available: https://docs.ros.org/en/jade/api/moveit_core/html/planning_scene_overview.html

[143] "Disinfection solution with UV-equipped mobile robots | OMRON, Europe." [Online]. Available: https://industrial.omron.eu/en/news-events/news/disinfection-solution-with-uv-equipped-mobile-robots

[144] A. Schmelzer and V. Miegel, "A Novel Approach To Advanced Cleaning In The Pharmaceutical Industry; 2 Worlds – 1 Robot," London, 4 2022.

[145] M. Beetz, L. Mösenlechner, and M. Tenorth, "CRAM - A Cognitive Robot Abstract Machine for everyday manipulation in human environments," *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pp. 1012–1017, 2010.

PUBLICATIONS RELATED TO THE THESIS

[WA1] A. Wolf and K. Szeéll, "A review on robotics in life science automation," in *Proceedings of the AIS 2019 14th International Symposium on Applied Informatics and Related Areas Organized in the Frame of Hungarian Science Festival*, 2019, pp. 106–111.

[WA2] A. Wolf, P. Galambos, and K. Szell, "Device integration concepts in laboratory automation," *INES 2020 - IEEE 24th International Conference on Intelligent Engineering Systems, Proceedings*, pp. 171–177, 7 2020.

[WA3] A. Wolf, D. Wolton, J. Trapl, J. Janda, S. Romeder-Finger, T. Gatternig, J. B. Farcet, P. Galambos, and K. Széll, "Towards robotic laboratory automation plug & play: The "lapp" framework," *SLAS Technology*, vol. 27, pp. 18–25, 2 2022.

[WA4] A. Wolf, S. Romeder-Finger, K. Széll, and P. Galambos, "Towards robotic laboratory automation plug & play: Survey and concept proposal on teaching-free robot integration with the lapp digital twin," *SLAS Technology*, vol. 28, pp. 82–88, 4 2023.

[WA5] A. Wolf, P. Zsoldos, K. Széll, and P. Galambos, "Towards robotic laboratory automation plug & play: Reference architecture model for robot integration," *SLAS Technology*, vol. 29, no. 4, p. 100168, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2472630324000505

[WA6] P. Zsoldos, A. Wolf, K. Széll, and P. Galambos, "Towards robotic laboratory automation plug & play: Lapp reference implementation with the tiago mobile manipulator," *SLAS Technology*, vol. 31, p. 100251, 2025. [Online]. Available: https://doi.org/10.1016/j.slast.2025.100251

[WA7] A. Wolf, S. Beck, P. Zsoldos, P. Galambos, and K. Széll, "Towards robotic laboratory automation plug & play: Lapp pilot implementation with the mobert mobile manipulator," in *2024 IEEE 22nd Jubilee International Symposium on Intelligent Systems and Informatics (SISY)*, 2024, pp. 000 059–000 066.

OTHER PUBLICATIONS

[WAO1] A. Wolf, P. Troll, S. Romeder-Finger, A. Archenti, K. Széll, and P. Galambos, "A benchmark of popular indoor 3d reconstruction technologies: Comparison of arcore and rtab-map," *Electronics 2020, Vol. 9, Page 2091*, vol. 9, p. 2091, 12 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/12/2091/htmhttps://www.mdpi.com/2079-9292/9/12/2091

[WAO2] A. Wolf, M. Paul, P. Galambos, and K. Szell, "Detecting gripping failure in a liquid handling robot with a break beam sensor," *SISY 2020 - IEEE 18th International Symposium on Intelligent Systems and Informatics, Proceedings*, pp. 101–106, 9 2020.

[WAO3] B. Parkinson, A. Wolf, P. Galambos, and K. Szell, "Assessment of the utilization of quadruped robots in pharmaceutical research and development laboratories," *INES 2023 - 27th IEEE International Conference on Intelligent Engineering Systems 2023, Proceedings*, pp. 221–228, 2023.

[WAO4] A. I. Cooper, P. Courtney, K. Darvish, M. Eckhoff, H. Fakhruldeen, A. Gabrielli, A. Garg, S. Haddadin, K. Harada, J. Hein, M. Hübner, D. Knobbe, G. Pizzuto, F. Shkurti, R. Shrestha, K. Thurow, R. Vescovi, B. Vogel-Heuser, A. Wolf, N. Yoshikawa, Y. Zeng, Z. Zhou, and H. Zwirnmann, "Accelerating discovery in natural science laboratories with ai and robotics: Perspectives and challenges," *Science Robotics*, vol. 10, no. 106, p. eadv7932, 2025. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.adv7932

Appendix A

SUPPLEMENTARY MATERIAL

S1 Results of the survey

See `Commands_survey.xlsx`.

S2 LAPP-RAP - Extended taxonomy

TABLE S1: Namspace

Namespace	**Description**
cleaning / CleaningController	Cleaning operations, e.g., area coverage with vacuum cleaner or UV desinfection robot [143, 144]
cleaning / SprayController	Spray a surface with a liquid, e.g., for desinfection
cleaning / WipeController	Wipe a surface, e.g., with a sponge [82]
devicemanipulation / BayController	Set a bay storage (e.g., carousel) to a desired position, so that a certain site can be accessed.
devicemanipulation / HatchController	Manipulate various hatches of devices, e.g., drawers, doors, sliding door, lids. Latching must be taken into account.
devicemanipulation / UIController	Interface with the physical user interface of a device, e.g., by push buttons, rocking switches, sliders, knobs etc.
humaninteraction / SpeechService	Human-robot interaction through text-to-speech. Let the robot communicate with the human through speech.
labwaremanipulation / CapController	Apply and remove screw caps: cap and decap.
labwaremanipulation / ClampController	Control the flow in a tube by applying a clamb externally.
labwaremanipulation / ConnectorController	Connect lines, e.g., with aseptic connectors.
labwaremanipulation / labelController	Robot labels a labware for simple marking e.g., with a marker. Plotter-like operation.
labwaremanipulation / LabwareTransferController	Pick-and-place labware transfer [84, 145]
labwaremanipulation / LidController	Lid and delid. Covers that are not screw-on. In its simplest form just place the lid over the container. In more complex cases the operation of some locking mechanism might be necessary, e.g., snap, latch or bayonet.
labwaremanipulation / LidFlipController	Open and close flip lids.
labwaremanipulation / PackagingController	Pack and unpack labware, remove or add single-use packaging. e.g., peal off packaging before loading the labware in a device, or pack it before shipment.
labwaremanipulation / SlideInController	Subtask-level pick and palce actions where the labware must be slid onto a platform when inserted into a device / site.
labwaremanipulation / TrolleyController	Move wheeled object, e.g., a trolley by docking to it or grasping its handle and pushing or pulling it.

labwaremanipulation / TubeHandlingController	Tube handling, e.g., loading into tube welder. Handle tangling tubes with advanced perception, e.g., tactile sensing and / or vision. Set-up-and-leave: tubes in predefined fixtures.
lowlevel / ArmController	Low-level (action primitive), not task-specific control of the robot arm. E.g., move rotation, move linear, approach.
lowlevel / BaseController	Low-level (action ptimitive), not task-specific control of the mobile base. e.g., navigat intermediary.
lowlevel / GripperController	Low-level (action primitive), not task-specific control of the gripper (end effector).
maintenance / BatteryController	Battery management, e.g., send to charge.
maintenance / ConfigurationController	Configuration management, e.g., change end effector tool.
maintenance / InitializationService	Reinitialize the robot
maintenance / MapService	Maintain the map of the premises and the base positions for robot navigation.
maintenance / PositionService	Maintain the device's position (e.g., nests).
maintenance / ProgramController	Exposes the ability to start, stop and pause pre-defined programs saved in the robot. For example: a hard-coded robot program.
maintenance / StatusProvider	Provide information about the health, availability and current activities of the robot.
maintenance / TeachingService	Teach robot positions.
maintenance / ValidationService	Validate a robot program in regard to collaborative requirements.
perception / BarcodeProvider	Scan a barcode with an on-board camera or barcode scanner.
perception / LiquidLevelProvider	Detect the liquid level in a container, e.g., by computer vision.
perception / ObjectDetectionProvider	Provide a known or unknowh object's pose by detecting it in a 2D or 3D image. Typical use case is grasp detection or object tracking.
perception / PhotoProvider	Take a photo with an on-board camera, e.g., for audit or remote troubleshooting purposes.
perception / PresenceProvider	Use a sensor to detect the presence of an object.
perception / ShapeProvider	3D reconstruction of a room or object. [WAO1]

samplemanipulation / PipetteController	The robot performs a liquid transfer with a pipette, (hand-held or integrated). [81]
samplemanipulation / PourController	The robot performs a liquid transfer by pouring from one container to another one.
samplemanipulation / ShakeController	The robot mixes a sample by performing the predefined periodic motion (e.g., rocking or shaking). Frequency is limited compared to vortex shaker devices.
samplemanipulation / StirController	The robot mixes a sample with a submerged tool.
samplemanipulation / VortexController	The robot places the sample on a vortex mixer device's platform to perform the shaking. Admittance or soft gripper is necessary.

S3 Sequence diagrams

The steps correspond to the prototype implementation of the LabwareTransfer RARs as presented in section 5.3. We use the uniform color coding, as introduced in section 2.3. It can be seen that a (T) Task (red) is manifested as the labware transfer *feature*, while (S) Subtasks (orange) are distinct SiLA *commands*, which are, in a general case, issued by the scheduler. (Q) Motion sequences (yellow) are implemented on the robot's middleware (ROS) and are not exposed directly via SiLA. In the figure, these are represented as self-calls on the components' side.

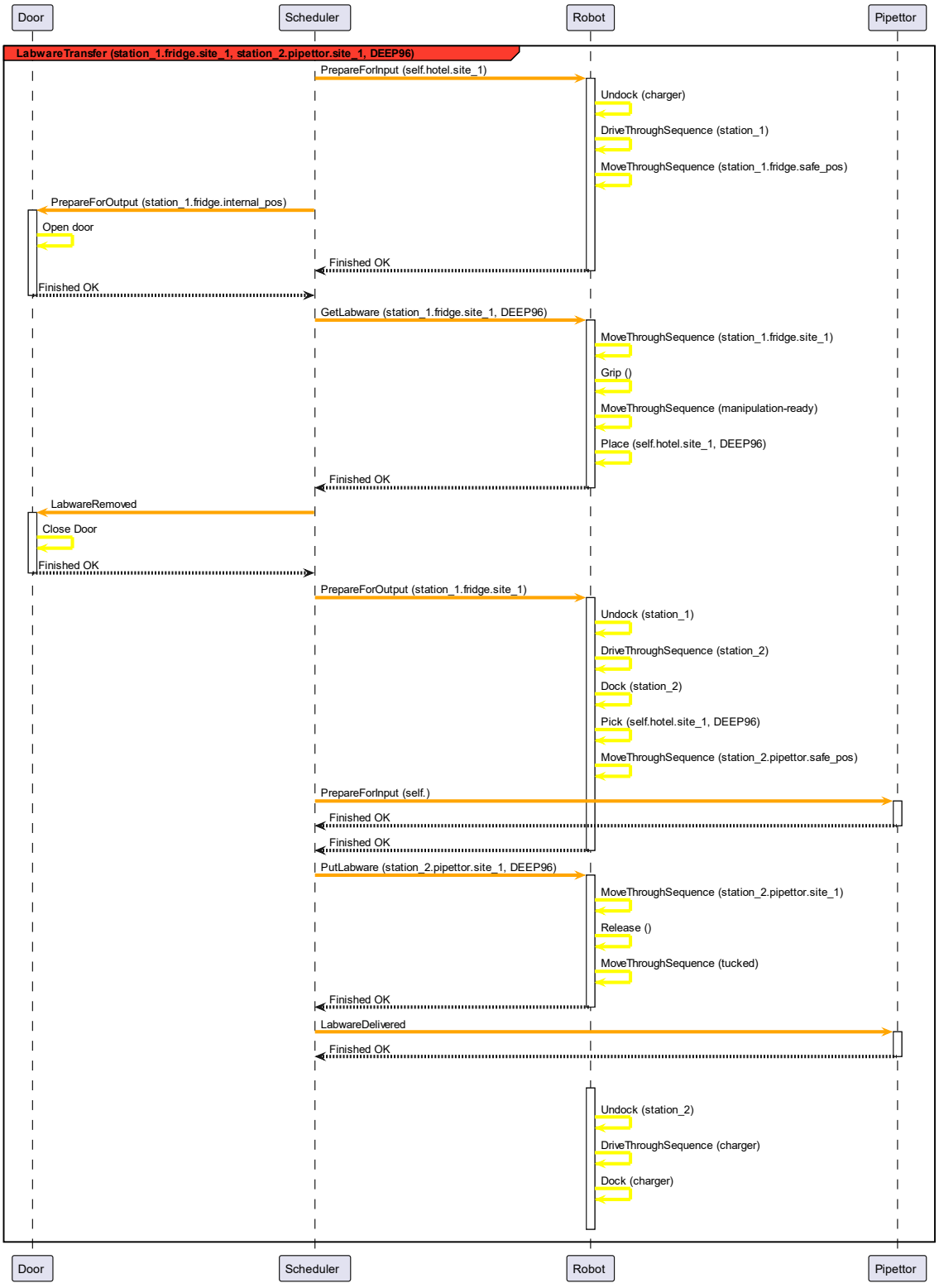


Fig. S1. Sequence diagram of a labware transfer task with a robot, between a fridge and a pipettor equipment - Part 1

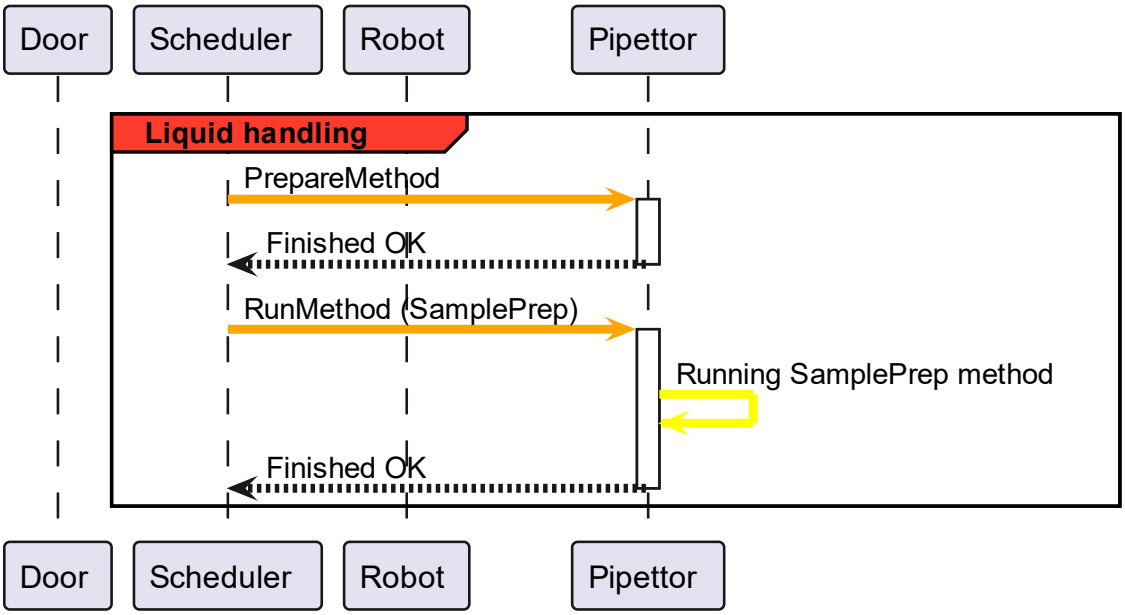


Fig. S2. Sequence diagram of a labware transfer task with a robot, between a fridge and a pipettor equipment - Part 2

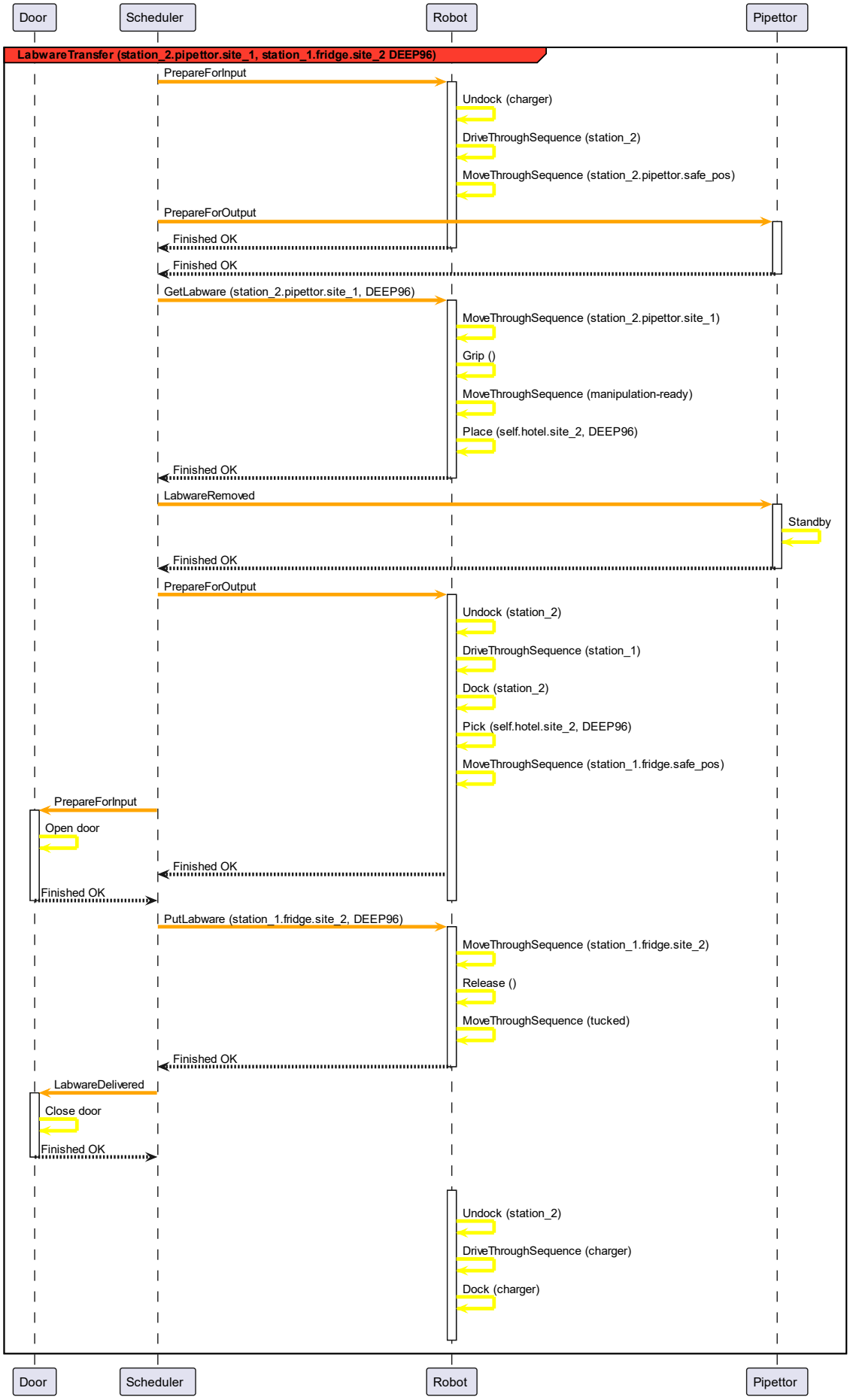


Fig. S3. Sequence diagram of a labware transfer task with a robot, between a fridge and a pipettor equipment - Part 3

S4 RAR mapping

Mapping our implementation to the Robotic Activity Representations (RARs) of the Laboratory Automation Plug & Play (LAPP) concept.
`TIAGo_RAR_sequence.xlsx`

S5 Test sequence video

Testing of the full sequence. View from the robot's head camera (top left), map view seen through RViz (top right), and two different recorded external views (bottom) displaying the physical robot.
`test_sequence_4x.mp4`