Technical University of Munich
TUM School of Engineering and Design

Integration platform for workflow automation and digitalization in bioprocess laboratories

Lukas Bromig

Complete reprint of the dissertation approved by the TUM School of Engineering and Design of the Technical University of Munich for the award of the

Doktor der Ingenieurwissenschaften (Dr.-Ing.).

Chair: Prof. Dr.-Ing. Harald Klein

Examiners:

1. Prof. Dr.-Ing. Dirk Weuster-Botz
2. Prof. Dr.-rer. nat. Marco Oldiges

The dissertation was submitted to the Technical University of Munich on 29 April 2025 and accepted by the TUM School of Engineering and Design on 20 October 2025.

Acknowledgments

I would like to thank Prof. Dr.-Ing. Dirk Weuster-Botz for the great support, mentorship, and academic guidance throughout this journey and for providing me with the opportunity to dive into the world of lab digitalization and automation.
An extended thank you to my colleague and friend Dr. Philipp Benner for all the discussions, input, and feedback, to Ingmar Polte for the late night companionship and scientific support, and to my project partner Dr. Nikolas von den Eichen for the practical and technical work, as well as the always interesting collaboration.
I am happy to have worked with great students - Maximilian Bochtler, Moritz Sturm, Valeryia Sidarava, Vladimir Romanenko, David Leiter - and am thankful for their contributions.
I would like to thank Prof. Dr. Anna-Lena Heins and Prof. Dr. Alberto Marin-Sanguino for their academic mentorship and Daniel Juchli and Dr. Matthias Arnold for their support, honest feedback, and for introducing me to the industry.
The Federal Ministry of Research and Education, I thank for their financial support within the framework of the project "DigInBio" (BMBF FKZ 031B0463B), and all project partners at FZ Jülich, Prof. Dr. Marco Oldiges, Prof. Dr. Wolfgang Wiechert, Dr. Michael Osthege, Dr. Johannes Hemmerich, and at the Leibniz University of Hannover, Prof. Dr. Thomas Scheper, apl. Prof. Sascha Beutel, Dr. Daniel Marquard, Dr. Marc Porr, Dr. Tessa Habich, for the scientific feedback and exchange at our frequent project meetings and beyond.
I am grateful to have my parents, brothers, friends, and especially Katharina. Their continuous support and reminders to find the time and motivation to work on and finish this thesis, next to my entrepreneurial endeavors, was invaluable.

Abstract

Industrial biotechnology is the application of biocatalysts, such as cells and enzymes, to industrial processes across the pharmaceutical, chemical, food, agricultural, materials, and energy sector. With the rapid technological advancements, especially within genetic engineering and synthetic biology, it has the potential to replace traditional processes with more economical and environmental friendly alternatives, while opening up a vast research exploration space, particularly in the pharmaceutical industry. As the technologies of biotech research and the resulting data is continuously increasing in complexity, cost and time efficiency becomes evermore important. To accelerate the development life cycle, from screening to bioprocess development and downstream processing, automation and digitalization is a necessary strategy to take advantage of these advancements.

Biotechnology and its applications are heterogeneous and numerous as the number of laboratory instruments and digital solutions on the market. Traditionally, software in the biotech space has been developed from a solution perspective, top-down for niche applications or with specific hardware in mind. The increasing requirements towards interconnectivity and thus integration of and between laboratory hardware and software is blurring the lines between vendor solutions. The resulting laboratory hard- and software ecosystems are as complex and rigid as they are non-interoperable from a software engineering and architecture perspective.

In this work, an integration platform inspired by industries such as manufacturing and business automation, was designed, adapted to the laboratory domain, and realized [Bromig *et al.* 2022a]. Supporting industry standards for system interfaces, a modular

approach to the laboratory software architecture is presented that enables more flexibility and faster implementation of bioprocess applications and experiments due to its generalizable and reusable nature.

The potential of this novel integration platform was demonstrated on two case studies: The first case study focused on the development of a dynamic, autonomous scheduling algorithm for a miniaturized and parallelized bioreactor system integrated into a liquid handling station. A static scheduling of process steps, such as substrate addition and pH-control, leads to long intervals between these actions and results in suboptimal process control and reduced yield. Parallelisation and dynamic prioritisation of these process recurring process steps enabled improved process control, reproducibility, and scale-up transferability as shown in the case study bioprocess in which the expression of a carboxylic acid reductase in recombinant *E. coli* was optimized [Bromig *et al.* 2022b; Von den Eichen *et al.* 2022].

The second case study accelerated adaptive laboratory evolution (ALE) experiments. In ALE experiments, microorganisms are cultivated in sequential batch processes in which they are exposed to an external selection pressure, such as cultivation on a non-native carbon source or the introduction of toxic substances. This leads to selection of sub-populations with advantageous mutations over multiple passages and thus an adaption to the selection pressure. An ALE experiment was automated and transferred to the L-scale, resulting in a overall acceleration by a factor of 9.4 as compared to the results published by the reference literature [Bromig *et al.* 2023].

Both case studies required interconnectivity between laboratory hard- and software systems across vendor barriers to implement the respective data flow and control logic. The integration of all laboratory hard- and software systems into the presented integration platform enabled the realization of the case studies based on a standardised and reuseable foundation and accelerated their implementation considerably. The

presented work showcases the benefits of introducing integration platforms into the laboratory software architecture and contributes to the advancement of automation and digitalization of bioprocess research laboratories.

Zusammenfassung

Industrielle Biotechnologie bezeichnet die Anwendung biologischer Katalysatoren, wie Zellen und Enzyme, auf industrielle Prozesse in den Bereichen Pharma, Chemie, Lebensmittel- und Materialwissenschaften sowie Energie. Mit den rasanten technologischen Fortschritten, insbesondere in der Gentechnik und der synthetischen Biologie, hat die industrielle Biotechnologie das Potenzial traditionelle Verfahren durch wirtschaftlichere und umweltfreundlichere Alternativen zu ersetzen und gleichzeitig ein großes wissenschaftliches Explorationspotenzial, insbesondere in der pharmazeutischen Industrie, zu eröffnen. Durch zunehmende Komplexität der Technologie und resultierender Daten, gewinnt Kosten- und Zeiteffizienz zunehmend an Bedeutung. Die Automatisierung und Digitalisierung von Arbeitsabläufen und Experimenten ist notwendig, um den Forschungszyklus von der Durchmusterungsphase über die Bioprozessgestaltung bis hin zur Produktisolierung zu beschleunigen.

Die Anwendungen der Biotechnologie sind ebenso heterogen und zahlreich wie die Vielzahl an Laborinstrumenten und digitalen Lösungen auf dem Markt. Traditionell wird Laborsoftware geräte- oder anwendungsspezifisch für Nischenanwendungen entwickelt. Die steigenden Anforderungen an Vernetzung und Integration zwischen einer Vielzahl an Laborgeräten und Laborsoftwaresystemen lässt Grenzen, besonders zwischen verschiedenen Herstellern verschwimmen. Die daraus resultierenden Geräte- und Softwarelandschaften sind aus software-architektonischer Sicht ebenso komplex und starr wie nicht interoperabel.

Im Rahmen dieser Arbeit wurde eine Integrationsplatform konzipiert und, inspiriert durch Lösungen aus der Fertigungsindustrie und dem Bereich der Geschäftsprozessautomatisierung, an die Anforderungen biotechnologischer Labore angepasst und umgesetzt [Bromig *et al.* 2022a]. Ergebnis ist eine modulare Laborsoftware-Architektur, die durch

Unterstützung moderner Industriestandards für Systemschnittstellen mehr Flexibilität und eine schnellere Implementierung von Bioprozessanwendungen und Experimenten ermöglicht.

Die Möglichkeiten dieser neuartigen Integrationsplattform wurde anhand von zwei Fallstudien demonstriert: In der ersten Fallstudie wurde eine dynamische, autonome Ablaufplanung für ein miniaturisiertes und parallelisiertes Bioreaktorsystem, das in einen Pipettierroboter integriert wurde, realisiert. Eine statische Ablaufsteuerung führt zu großen Zeitintervallen zwischen Prozessschritten, wie der Substratzugabe oder der pH-Kontrolle, was wiederum zu einer suboptimalen Prozessführung und reduzierten Produktausbeute führt. Die Parallelisierung und dynamische Priorisierung dieser Prozessschritte führt zu einer verbesserten Prozesskontrolle, Reproduzierbarkeit und Übertragbarkeit auf größere Maßstäbe. Dies wurde am Beispiel eines Bioprozesses gezeigt, in dem die Expression einer Carboxylsäure-Reduktase in rekombinantem *E. coli* optimiert wurde [Bromig *et al.* 2022b; Von den Eichen *et al.* 2022].

In der zweiten Fallstudie wurde ein Experiment zur adaptiven Laborevolution (ALE) automatisiert. Bei ALE Experimenten werden Mikroorganismen in aufeinanderfolgenden Satzverfahren einem Selektionsstress ausgesetzt, beispielsweise durch die Kultivierung auf einer nicht-nativen Kohlenstoffquelle oder die Zugabe toxischer Substanzen. Dies führt zur Selektion von Subpopulationen mit vorteilhaften Mutationen und damit über mehrere Passagen hinweg zur Anpassung an den Stress. Ein solches ALE-Experiment wurde automatisiert und auf den Liter-Maßstab übertragen, was eine Beschleunigung des Experiments um den Faktor 9,4 im Vergleich zur Referenzliteratur ermöglichte [Bromig *et al.* 2023].

Beide Fallstudien erforderten die Vernetzung von Laborgeräten und Laborsoftware über die Grenzen der Herstellerlösungen hinaus, um die entsprechende Logik für den Daten- und Kontrollfluss umzusetzen. Die Einbindung aller Geräte und Laborsoft-

waresysteme in die vorgestellte Integrationsplattform ermöglichte die Realisierung der Fallbeispiele auf einem standardisierten, wiederverwendbarem Fundament und beschleunigte deren Implementierung. Diese Arbeit zeigt die Vorteile von Integrationsplattformen und deren Anwendbarkeit als Grundlage für eine moderne Laborsoftwarearchitektur auf und trägt zur Weiterentwicklung der Automatisierung und Digitalisierung von Bioprozessforschungslaboren bei.

Contents

1 Introduction

Traditional laboratory work is undergoing a transformation, started by the digital revolution and fuelled by the rapid technological advances in the field of biotechnology. Manual processes and paper-based documents and data are turning digital.

While manual work is already being replaced with automated island solutions, the high costs associated prevent broader adoption beyond typical high-throughput use cases. Dedicated systems for specific application areas and vendors, ranging from liquid handling to bioreactor system control, have led to an explosion of software products. Missing interoperability across vendor software and hardware is caused by a lack of interface standardization.

Apart from physical automation, the digital transformation journey of companies in the life science domain starts with the digitalization and automation of their data and data flow. The product of research is data and thus data acquisition, processing, and analysis is typically the starting point for any digital transformation journey. Traditional data acquisition consists of lab notebooks, distributed excel sheets and exports in various file types and formats from analytical instruments. Processes for the central acquisition, homogenization, and storage of data based on the digitalization of the status quo is prioritized over the transformation of the underlying processes of data and meta data generation themselves. Flat files are not required if systems can exchange data directly with each other.

Successful digitalization requires both, physical process automation and data automation. To achieve this, overarching concepts and software infrastructure are required that support flexible adaption as processes and the science behind them evolve. The current software landscape in the lab is highly fragmented, isolated, and outdated.

Vendor-specific solutions cover the entire value chain from the building of hardware to instrument integration to the corresponding closed software ecosystems. This status quo of vertical integration is unscalable, rigid, and prohibitively expensive for the majority of use cases: low- and medium-throughput processes. A chaotic landscape of disconnected hard- and software solutions grows organically. Beyond maintaining them, integrating these systems with each other, is one of the biggest hurdles of the industry and one of the highest cost factors of any automation and digitalization effort.

The laboratory of the future is evolving into an interconnected, automated, and highly efficient ecosystem of modular software applications, transforming the way science is conducted. This shift is driven by the convergence of robotics, automation, compute and storage availability, and the establishment of robust standards for data structures and system connectivity [Beutel *et al.* 2022; Habich *et al.* 2024; Zupancic *et al.* 2021]. Collectively, these advances improve data integrity, quality, and reproducibility – pillars crucial to the scientific process in an era increasingly dependent on and preparing for artificial intelligence.

System connectivity, accessibility, and modularity are cornerstones of this transformation. By enabling seamless communication between instruments, it can be ensured that data collection is comprehensive, accurate, and structured from the moment of acquisition. Only interconnectedness provides meta-data that is otherwise lost in between process steps. This addresses long-standing issues like the reproducibility crisis in scientific research by fostering consistency in experimental methodologies [Zupancic *et al.* 2021]. The adoption of interoperability standards, such as Standardization in Laboratory Automation (SiLA 2) and Open Platform Communications Unified Architecture (OPC UA), simplify device integration, breaking down data silos and enabling centralized control of diverse laboratory equipment [Beutel *et al.* 2022]. An industry-wide consolidation of the integration effort based on these standards will have a far reaching

impact on the cost, speed, and quality of laboratory work.

Automation lies at the heart of the smart laboratory, replacing labor-intensive processes with robotic systems capable of executing complex workflows with precision and speed. As processes and experiments, such as transcriptomics and next-generation sequencing (NGS) workflows, evolve, automated systems not only minimize human error, but make them possible in the first place. Opposed to the status quo of manual processes, in which data need to be processed and moved to be made FAIR (Findable, Accessible, Interoperable, Reusable), automated processes can be set up to generate data that is inherently "born FAIR", enriched, and cross-referenced with information that supports AI-driven analyses and predictive modeling [Zupancic *et al.* 2021]. Scientific and experimental work will shift toward experimental planning, data modeling, simulation, and optimization and away from physical laboratory work [Arnold 2022; Beutel *et al.* 2022].

Cloud computing and secure digital infrastructures play pivotal roles in managing the exponential growth of scientific data [Bloomberg 2016]. By decentralizing storage and computational resources, these technologies provide scalable, secure, and accessible platforms for data sharing and collaboration. They also support compliance with regulations like General Data Protection Regulation (GDPR) when handling sensitive biological data or with Good Laboratory Practices and Good Manufacturing Practices (GxP) regulation and guidance that focuses on audit trails and software validation, ensuring that the lab of the future operates within ethical, safe, and legal boundaries [ISPE | International Society for Pharmaceutical Engineering 2022; U.S. FDA 2024; Zupancic *et al.* 2021].

With data and many process management systems moving to the cloud, the amount of physical and data automation logic deployed locally in the laboratory, the traditional desktop application, or even "do it yourself" (DIY) solutions are becoming increasingly

problematic [Steinwandter *et al.* 2019]. Concepts from the manufacturing industry, including the convergence of information technology systems and operational technology systems (IT/OT) or the Administration Asset Shell (AAS), a form of digital twin, are slowly introduced into the laboratory space [Arm *et al.* 2021; Ehie *et al.* 2020; Giannelli *et al.* 2022].

The impact of digitalization is especially profound in microbiology and bioprocessing. In microbiology, multi-omics approaches and advanced bioinformatics tools are unlocking microbial diversity, facilitating the discovery of robust enzymes and biocatalysts for sustainable industrial processes [Abramson *et al.* 2024; Helleckes *et al.* 2023a; Wolf *et al.* 2018]. In bioprocessing, current trends and published tools encompass precise modeling of biological systems, process optimizaion with closed-loop bayesian optimization approaches, and real-time monitoring of production systems through soft sensors and machine learning algorithms [Helleckes *et al.* 2023a; Krüger *et al.* 2020; Rathore *et al.* 2022]. The integration of digital twins in bioprocessing enhances predictive capabilities, streamlines operations, and contributes to the development of eco-friendly, zero-waste production technologies, which are essential for transitioning to a bio-based circular economy [Krüger *et al.* 2020]. However, incorporating these technologies into the laboratory environment and establishing connectivity for control and data access to laboratory hard- and software poses a time-consuming and prohibitive challenge that remains unsolved in a generalizable way. Every stakeholder in the field of bioprocess research, from hardware and solution providers to academic and industrial laboratories, integrate the same hardware systems from scratch into a plethora of custom software solutions over and over again.

The ultimate vision for this smart biotechnology lab is a "self-driven" system, where experiments are autonomously designed, executed, and analyzed by specialized, interoperable software applications and AI agents, supervised and directed by a human

researcher [Future House Inc. 2024; Laurent *et al.* 2024; Sim *et al.* 2024; Skarlinski *et al.* 2024; Tom *et al.* 2024]. This paradigm reduces human labour, allowing scientists to focus on creative and strategic aspects of research. By prioritizing data quality, system integration and interconnectedness, as well as automation, the lab of the future promises to revolutionize scientific discovery, making it more agile, efficient, and impactful.

2 Motivation

2.1 Challenges in Laboratory Automation and Digitization

Laboratory automation has long promised increased efficiency, accuracy, and throughput in biotechnology research, yet its adoption in academic and small-scale settings remains slow and uneven [Jessop-Fabre *et al.* 2019]. Many experimental workflows are still executed manually, leading to issues with reproducibility and labor-intensive processes [Holland *et al.* 2020]. A core challenge is the fragmented software landscape in laboratories: researchers often rely on a patchwork of specialized, vendor-specific software tools that do not communicate with each other. When instruments and software systems operate in isolation, it creates inefficiencies – data must be transferred by hand, transcription errors creep in, and workflows become error-prone and difficult to scale. Integrating new automation technology with existing lab equipment and software is a common point of failure. These integration issues not only reduce efficiency but also make it harder to maintain data integrity and focus on the research topic at hand.

Another significant problem is the high barrier to entry for automation in research laboratories. Commercial automation solutions tend to be expensive, inflexible, and tailored to industrial environments, putting them out of reach for many academic labs with limited budgets [Holland *et al.* 2020; Jessop-Fabre *et al.* 2019]. Traditional laboratory robots, liquid handling stations, and automation software often require specialized expertise to operate and months of setup time, which discourages their widespread adoption in agile research settings. Moreover, each lab's protocols can be highly specialized or frequently changing, and rigid automation systems struggle to accommodate

such variability. As a result, researchers who lack programming skills or substantial funding face a steep challenge in implementing automation. In many cases, labs are "forced into developing their own systems" from scratch to meet their needs [Holland *et al.* 2020]. This status quo has led to duplicated effort and a proliferation of incompatible one-off solutions across the biotechnology community, further exacerbating the fragmentation problem. The slow pace of automation uptake in academia – described as an "automation gap" – can be attributed to these combined technical and resource challenges [Holland *et al.* 2020].

2.2 Need for Reusable and Modular Automation Infrastructure

Addressing these challenges requires reimagining how we design lab automation software. There is a clear need for reusable, modular, and standardized automation infrastructure that lowers the threshold for adoption. Researchers have argued that increasing automation in research labs will "require more flexible, modular and cheaper designs" to suit laboratories with constrained resources [Holland *et al.* 2020]. Instead of monolithic systems or ad-hoc custom scripts, a modular approach would allow automation workflows to be composed of interchangeable components that can be easily added, removed, or reconfigured as needs evolve. Such flexibility helps prevent obsolescence when protocols change, and it minimizes redundant functionality. Equally important is making these tools accessible: open architectures and open-source software can empower scientists who lack extensive engineering expertise. Open-standard and open-source solutions have an enabling effect for researchers without programming skills, lowering barriers to entry and cost. By sharing device interfaces and protocol code, the scientific community can avoid reinventing the wheel for each new project.

Standardization is a key enabler of reuse. In the current landscape, the lack of common interface standards means one lab's automation cannot easily be transferred to another's setup without significant rewriting. This has driven interest in adopting interface standards that define unified ways for software to communicate with laboratory instruments. For example, the SiLA (Standardization in Lab Automation) consortium has developed SiLA 2 as an open standard for instrument interfaces. "SiLA's mission is to establish international standards which create open connectivity in lab automation" and enable "interoperability, flexibility, and resource optimization for laboratory instrument integration" [Juchli 2022]. SiLA 2 provides a modern, networked communication protocol for devices, emphasizing a common vocabulary and communication layer that is independent of any specific vendor [Jessop-Fabre *et al.* 2019; Juchli 2022]. Using such standards can fundamentally improve interoperability: devices from different manufacturers that implement the standard can be integrated into workflows with minimal custom coding. In essence, standardized interfaces turn formerly siloed instruments into plug-and-play modules of a larger automated ecosystem. This greatly reduces the engineering effort needed to add or swap out lab equipment, which is particularly beneficial for small laboratories that incrementally build out their automation capability.

2.3 Integration Platforms and the SiLA 2 Approach

While interface standards define how devices communicate, laboratories also require integration platforms to coordinate multiple instruments and connect them to higher-level systems such as Laboratory Information Management Systems (LIMS) or Electronic Laboratory Notebooks (ELN). These platforms serve as a middle layer that separates device-level integration from application-specific experiment logic. By abstracting instruments into modular, reusable components, integration platforms allow researchers

to design complex workflows without rewriting code for each individual device.

The SiLA 2 standard plays a central role in this process by enabling uniform device control through standardized interfaces. An integration platform based on SiLA 2 can manage device discovery, communication, and execution, while providing an independent layer for experiment logic and scheduling. This decoupling greatly enhances software reusability and flexibility: instruments can be replaced or added with minimal changes to workflow code, and new protocols can be developed independently of the hardware used.

2.4 Research Goal and Approach

Given the above considerations, the overarching goal of this research is to enable generalizable automation in biotechnology laboratories through a modular integration platform that decouples instrument integration from application logic. In pursuit of this goal, the work focuses on designing a software architecture (referred to as the SiLA 2 Manager) that serves as an integration hub for laboratory devices, using the SiLA 2 standard as the backbone for communication. This platform is intended to make device integration sustainable and fast – once a new instrument is wrapped with a SiLA 2 interface, it can be plugged into the platform and immediately made available for any automated workflow, without further custom coding. By leveraging the SiLA 2 standard's open and self-describing interface structure, the platform ensures that adding a new device or replacing an old one does not require reengineering the rest of the system. The application logic (the experimental workflow definitions, decision rules, and scheduling applications) is developed independently on top of this device layer. This means researchers can focus on the scientific protocol, while the platform handles the low-level device interactions. The approach thereby lowers the technical barrier for laboratories to implement automation: a researcher can compose complex experimental routines

by reusing existing device modules and writing high-level logic in a modular fashion, instead of building an entire system from scratch for each new project.

The platform enables the execution of complex experimental workflows by cleanly separating device integration from experimental logic. It provides consistent access to a variety of laboratory instruments, allowing researchers to develop control strategies that respond to real-time data and coordinate actions across devices. This architecture simplifies the development of workflows that involve time-sensitive tasks, conditional logic, and coordinated scheduling. By reducing the need to rewrite integration code for each experiment, the platform supports rapid development and adaptation of automated processes across diverse applications.

2.5 Objectives

This cumulative dissertation is composed of three peer-reviewed publications, each addressing a specific facet of the overarching goal:

2.5.1 Laying the foundation: Developing a device-agnostic integration platform

The first publication presents the design and implementation of the SiLA 2 Manager platform. The objective was to create a general integration platform that uses the SiLA 2 interface standard that allows sustainable and rapid integration of laboratory devices. The publication details the platform's architecture, which encapsulates devices as SiLA 2 services and, due the to self-describing nature of the interfaces, enables the addition of new instruments with minimal effort. The result is a reusable integration infrastructure that any compatible device can hook into, thereby reducing the time and software development work needed whenever a new instrument is introduced into the lab

or is required for a new biological use case. This platform lays the groundwork for the subsequent application-focused studies and was validated on a bioprocess optimization case study.

2.5.2 Enabling complex workflows: Dynamic scheduling for parallel bioprocess control at mL-scale

The second publication applies the integration platform to a complex, multi-device setup involving a parallelized mL-scale bioreactor system operated via a high-throughput liquid handling station. The goal was to demonstrate how dynamic process control could be achieved by orchestrating devices from different vendors using the same modular infrastructure. A dynamic scheduling algorithm was implemented to coordinate tasks such as substrate addition, sampling, and at-line analytics across 48 parallel bioreactors. Building on the reusable integration components from the previous work, only the dynamic scheduling logic specific to this use case had to be developed, highlighting how the infrastructure reduces development effort and enables flexible automation.

2.5.3 Demonstrating reuse: Automating adaptive laboratory evolution at L-scale

The third publication validates the generalizability and scalability of the approach by applying the same integration platform to a very different application: adaptive laboratory evolution (ALE) experiments at L-scale. ALE is a long-term, adaptive process requiring continuous culture of microorganisms with periodic interventions (feed adjustments, sampling, passaging) based on growth dynamics. The goal was to automate ALE in L-scale bioreactors using the same software infrastructure, to prove that one can transition to a new experimental domain with minimal redevelopment. The only major addition needed was new application logic specific to the ALE experiment (for

instance, criteria for automated passaging and a soft-sensor for biomass estimation), while the underlying device integrations were directly reused from the existing platform. The publication presents how, by reusing the SiLA 2 Manager and its device drivers, the application for an automated ALE setup controlling bioreactors, pumps, and sensors, was rapidly developed. This case underscores the reusability of the platform: even though the scale (from mL to L) and the experimental goals were different, the integration framework did not require rewriting, confirming that the decoupling of integration from logic can dramatically reduce redundant software development when moving between projects.

2.6 Reusability, Sustainability, and Scientific Relevance

The architectural philosophy championed in this work – modular integration with standardized interfaces – carries significant benefits for both software sustainability and scientific productivity. By designing automation software as a collection of reusable modules, we ensure that new projects can build on prior work instead of starting anew each time. This greatly reduces redundant development: for instance, a device driver or control routine written once (as a SiLA 2 service in the platform) can be employed in many different workflows. In contrast, without such infrastructure, laboratories often end up rewriting similar instrument integration code for each experiment or resorting to entirely manual operation if the coding burden is too high. The presented approach demonstrates a path away from this inefficiency. It creates a sustainable software ecosystem for the lab, where adding capabilities is incremental and cumulative. Over time, the library of device integrations and protocol modules grows, enabling faster setup of future experiments and fostering sharing of methods between researchers. This aligns with the broader trend in the scientific community towards open, modular toolkits that can be extended and repurposed rather than closed, one-off systems.

Scientifically, the ability to rapidly prototype and iterate complex experiments using automation can accelerate discovery. In Publication 2, the dynamic optimization of bioprocess conditions is an example of an experiment that benefits from automation not just for labor-saving, but for exploring a larger experimental space with better control than a human could achieve. Similarly, the ALE study (Publication 3) would be logistically very challenging to perform reliably without automation due to the long timescales and frequent interventions required. By making these kinds of studies easier to implement, the platform contributes to scientific progress in biotechnology – experiments can be more ambitious, more reproducible, and more data-rich. Moreover, the generalizable nature of the platform means that insights and methods developed in one context (e.g. bioprocess optimization) can be translated to another (e.g. evolutionary engineering) with minimal friction, encouraging cross-pollination of techniques. This reusability is not only a software advantage but also a scientific one: it lowers the barrier to trying innovative automated approaches in different subfields of biotechnology.

From an architectural perspective, this work also serves as a reference model for lab automation software design. It highlights how adherence to interface standards and clean separation of concerns can yield software systems that are both flexible and robust. The SiLA 2 Manager exemplifies how to implement a standard like SiLA 2 in a real integration context, potentially guiding other developers or laboratories aiming to adopt SiLA 2 for their own automation. In that sense, the dissertation's contributions are as much about establishing best practices as about delivering specific experimental results.

3 Theoretical Background

The following sections provide an overview of the current laboratory software landscape and various standardization efforts. To contextualize this work and explore future trajectories for the digitalization of biotechnology - particularly in laboratories - it is essential to examine the status quo in other industries, such as manufacturing and business automation. Standardization and formalization efforts are key to creating a more interconnected and interoperable software landscape and are therefore a central focus.

3.1 Laboratory software

Existing software solutions like Electronic Lab Notebooks (ELNs) and Laboratory Information Management Systems (LIMS) help acquire unstructured and structured data, respectively, but often lack the flexibility or device access needed to support a wide variety of applications. Scientific result data is the product of laboratory work, and most software systems in the lab focus on data generation, acquisition, and overarching processes to ensure reproducibility and data integrity. Major promises to implement software systems in the lab include the reduction of errors and time savings [Kranjc 2021].

In most cases, software systems are domain and even application specific. A quality control lab of a pharmaceutical production site has different requirements for a LIMS than a research facility in drug development. At the same time, an application-specific software to facilitate liquid handling operations is vastly different from a software for bioprocess control. A typical lab environment consists of multiple software solutions and as research processes start stretching across vendor and solution barriers and requirements for data integrity, metadata and process data increase, connectivity between

these software systems becomes increasingly important [Biermann *et al.* 2021; Seidel *et al.* 2022].

Integration platforms provide the glue between these software systems by integrating and homogenizing proprietary interfaces of laboratory instruments and laboratory software, and by providing an environment to deploy research process logic that connects them. This can include data pipelines for data acquisition and transformation or even bioprocess control logic. The integration platform stretches across the laboratory, thus bridging the gap between the digital world and physical laboratory.

However, integration platforms require substantial resources for training, custom integration, and implementation. For this reason, integration platforms often resort to file system integration - that is, a shallow integration that does not connect directly to the instrument and therefore cannot be used for physical automation.

The integration of hardware and software systems in life science laboratories is vital for the digital transformation. Laboratories today are advancing towards automation and interconnectedness to increase process efficiency, reproducibility and improve data utilization. However, significant challenges, including a fragmented software system landscape as well as a lack of universal data, interface, and process standards, impede this progress. To overcome these barriers, broader adoption of standards, along with investment in scalable technologies, is required [Biermann *et al.* 2021; Scheitz *et al.* 2018; Seidel *et al.* 2022].

Prevalent laboratory software systems, the technological evolution of similar software systems in adjacent industries, inherent challenges of digitalization, and emerging solutions, with a focus on integration platforms and industry standards, are presented in this section.

3.1.1 Software systems in life science laboratories

Laboratories are fraught with research process inefficiencies that hinder productivity and increase the risk of errors. These inefficiencies stem from manual data handling, siloed data in disconnected file systems and databases, and complex workflows. One major challenge is the reliance on manual data entry and transfer between disparate systems, such as from analytical instruments to LIMS. This process often introduces transcription errors, delays, and inconsistencies. These issues not only slow workflows but also risk compromising research integrity. The prevalence of manual data transfer processes underscores the urgent need for integrated software solutions that enable seamless and automated data flow.

Disjointed workflows are another key challenge resulting from disconnected laboratory software systems that require researchers to navigate between multiple graphical user interfaces, typically one per instrument. This fragmentation disrupts workflows and reduces productivity, as researchers spend valuable time coordinating data flow and running operations manually. For instance, robotic liquid handlers often operate independently of scheduling or data integration solutions, necessitating manual oversight to ensure smooth execution. Such inefficiencies underscore the need for connected and interoperable systems that can harmonize device operations and data exchange.

The lack of robust sample tracking and comprehensive documentation creates further inefficiencies and risks in laboratory operations. Poor traceability often results in misplaced or mislabeled samples, making it difficult to reproduce results reliably or backtrack irregularities. This issue is particularly acute in regulated environments where non-compliance with documentation standards can result in halted operations or jeopardized certifications. Software solutions that integrate audit trail features, inherent to automated solutions, are essential to mitigating these risks and ensuring operational continuity.

Limited automation capabilities in many laboratories exacerbate inefficiencies, especially in high-throughput environments. Legacy systems, often unable to support modern automation, slow down processes and place additional burdens on human operators. The absence of automation not only increases the risk of errors but also limits the capacity for scaling operations to meet growing research demands. Modern laboratory execution systems (LES) and integrated robotic solutions are critical to bridging this gap by enabling automated, reproducible workflows.

Finally, data silos and accessibility issues hinder real-time analysis and decision-making. Data generated by various laboratory instruments is frequently stored in incompatible formats or isolated data management systems, making it challenging to aggregate and analyze holistically. This fragmentation obstructs efforts to implement FAIR data principles, which are crucial for collaborative and reproducible research. Data integration solutions are the current solution for managing data flow, harmonizing formats and structure, and enabling comprehensive insights across laboratory operations.

Life science laboratories employ domain specific software systems to manage processes, data flow, and physical automation. Key systems include:

Laboratory Information Management Systems (LIMS)

LIMS, such as LabWare LIMS[1] and STARLIMS[2], are fundamental for managing structured laboratory processes. They emphasize sample tracking and process monitoring, inventory management, and audit trail generation to ensure data integrity. LIMS are process-driven and produce structured, predefined data outputs. They often incorporate features from other software categories, such as ELNs and LES, creating multifunctional

[1]LabWare, Inc., Wilmington, Delaware, USA

[2]STARLIMS Corporation, Hollywood, Florida, USA

one-size-fits-all ecosystems. Direct connections to devices or file systems are typically not included. Although LIMS are designed for the automation of defined processes, such as a defined sample preparation method followed by a dedicated assay, they primarily focus on digitalizing these and generally do not include any physical automation in the laboratory. However, direct device connectivity, such as instrument parameterization or reading results from instruments (e.g., weighing data), is becoming increasingly important for providing seamless solutions.

Electronic Lab Notebooks (ELNs)

ELNs, like LabForwards Labfolder[3] or Benchling ELN[4], replace traditional laboratory notebooks with digital ones that emphasize collaboration and data traceability. ELNs do not follow a strictly structured process and are used in a more explorative and flexible research context, mostly for experiment documentation. The resulting data is unstructured. ELNs often coexist with LIMS and LES software and a direct connection to devices or connected file systems is becoming increasingly important, mostly for data integrity and time saving purposes, however rarely existent.

Laboratory Execution Systems (LES)

LES software, such as LabOperator by LabForward[5] or Artificial's LABOPS[6], ensure protocol compliance and enable the automation of experimental procedures, to ensure reproducibility and data integrity. A direct connection to devices or connected file systems for execution is required. Many ELN and LIMS software systems have inbuilt LES features to guide operators in the laboratory through digital standard operating procedures (SOPs) in a step-by-step manner. LES are used in human and semi-automated human-in-the-loop processes.

[3]labforward GmbH, Berlin, Germany

[4]Benchling, Inc., San Francisco, California, USA

[5]labforward GmbH, Berlin, Germany

[6]Artificial, Inc., Palo Alto, California, USA

Scheduling Software

Most physical, full automation is done by custom built workcells. These workcells consist of a set of laboratory instruments that, if combined, can run a complete workflow, like a sample preparation, an ELIZA assay, cherry-picking, or a library preparation, independently. Workcells are operated by scheduling software of the respective workcell or hardware vendor. Software solutions like Green Button Go Scheduler[7] and Cellario-Scheduler[8] optimize resource allocation and coordinate robotic systems in application-specific workcells. These types of schedulers can parallelize multiple workflows and adjust schedules to react to feedback and are therefore referred to as dynamic schedulers. The solution providers develop low-level device drivers to connect to the involved instruments. As isolated "island solution", they also require integration into higher-level systems such as LIMS or data integration platform solutions.

Data Integration Solutions

Data integration solutions, including Scitara[9], TetraScience[10], Ganymede[11], and Zontal[12], focus on automating data flow from the laboratory to higher-level data management systems such as LIMS, data lakes, and analysis platforms. Their core concept relies on file system agents in the lab that acquire generated data and trigger an extract-transform-load (ETL) pipeline. This pipeline harmonizes the data format and structure and ultimately stores it in a central location. However, these solutions lack direct automation capabilities and integrate only with the computer file systems connected to the instrument. Direct connections to the instrument for data acquisition purposes

[7]Biosero, Inc., San Diego, California, USA

[8]HighRes Biosolutions, Inc., Beverly, Massachusetts, USA

[9]Scitara Corporation, Marlborough, Massachusetts, USA

[10]TetraScience, Inc., Boston, Massachusetts, USA

[11]Ganymede Bio, Inc., Palo Alto, California, USA

[12]ZONTAL, Inc., Provo, Utah, USA

(e.g., Industry of Things (IoT) enabled use cases) are rarely implemented.

The overlapping functionalities of these systems underscore the need for integration and standardization to achieve seamless workflows and data sharing. While data integration solutions solve many of the current needs of lab connectivity, future applications will require a deeper integration level, on the same level as scheduling applications [Biermann *et al.* 2021].

Furthermore, the largest group of software systems in the lab consists of vendor and device specific desktop applications. While many of the aforementioned systems provide application programming interfaces (APIs), typically REStful APIs, instrument vendor software is particularly difficult to integrate.

3.1.2 The evolution of data integration platforms across industries

The necessity for interface standardization and robust software infrastructure has long been established in industries like manufacturing, retail, and financial services. Integration platforms in these domains provide frameworks to connect disparate systems, enabling automation and data centralization. Examples include:

MuleSoft Anypoint Platform[13]: Widely adopted in finance, retail, and healthcare for API-led integration and seamless data sharing.

Tulip Interfaces[14]: Used in manufacturing to digitize shop floor operations, bridging hardware and software for productivity enhancements.

Zapier[15]: A no-code platform popular in marketing and sales for automating workflows

[13]Salesforce, Inc., San Francisco, California, USA

[14]Tulip Interfaces, Inc., Somerville, Massachusetts, USA

[15]Zapier, Inc., Columbia, Missouri, USA

across web applications.

Airbyte[16] and Fivetran[17]: Tools for data integration and Extract, Transform, Load (ETL) processes, aiding companies in analytics and e-commerce by centralizing data sources.

UIPath[18] and Workato[19]: Leaders in process automation, these platforms optimize workflows in sectors like customer service and insurance.

These integration platforms exemplify the role of integration solutions in simplifying complex workflows, enhancing interoperability, and fostering automation. Abdou *et al.* [2021] provides additional examples and a comparative study of integration platforms in these industries. The key differentiator between these aforementioned solutions, compared to the laboratory environment, is the lack of physical systems that need to be integrated and that they operate in a space in which RESTful APIs have been established as the status quo interface.

Examples in manufacturing, the industry closest to the laboratory field, include PTC ThingWorx[20], which is well established in industrial IoT (IIoT) environments [Biermann *et al.* 2021]. It is designed for monitoring and data aggregation through their proprietary AlwaysOn protocol, which is based on Websocket technology. However, it is not used for device control required for robotic, physical process automation or for the transmission of large data packages (e.g., in microscopy or NGS applications) as encountered in the laboratory environment.

Similar IIoT platforms in the life science industry, such as the Elemental Machines

[16] Airbyte, Inc., San Francisco, California, USA

[17] Fivetran, Inc., San Francisco, California, USA

[18] UI Path, Inc., New York, New York, USA

[19] Workato, Inc., Mountain View, California, USA

[20] PTC, Inc., Boston, Massachusetts, USA

Platform[21], focus on IoT-enabled laboratory monitoring of environmental sensors, such as refrigerators, freezers and incubators, but also lack capabilities for directly controlling laboratory instruments, especially more complex hardware, like bioreactors or liquid handling stations, or transmitting large amounts of data.

A software infrastructure for flexible and sustainable laboratory digitalization and automation requires not just data access, but also instrument control. Pallasch *et al.* [2018] have highlighted these requirements for the manufacturing industry and Biermann *et al.* [2021] have presented insights into the challenges and shortcomings when translating these requirements to the laboratory industry in particular.

3.2 Data standardization

3.2.1 Data integrity, quality, and quantity

Data integrity and quality are fundamental to scientific research, yet challenges persist due to non-FAIR data practices. Such practices impede collaboration and innovation by rendering datasets difficult to share and reuse across various research contexts. For instance, data stored in incompatible formats or fragmented systems creates silos, hindering researchers from building upon existing work. This lack of interoperability contributes to the reproducibility crisis in science, where findings cannot be replicated due to incomplete, poorly structured, or inaccessible data [Neumann 2022; Russell 2021; Wilkinson *et al.* 2016]. Economically, these inefficiencies are significant, with the European economy losing an estimated €10.2 billion annually due to suboptimal data practices, alongside other unquantified losses [for Research and Innovation (European Commission) *et al.* 2018].

The roots of poor data quality are multifaceted. A major issue is the reliance on pro-

[21] Elemental Machines Inc., Cambridge, Massachusetts, USA

prietary systems that hinder interoperability. In addition, manual data entry and insufficient validation processes introduce errors and inconsistencies. Another significant problem is the absence of comprehensive metadata that provides essential context, such as experimental conditions, data provenance, and instrument information. Without robust metadata, datasets are incomplete, limiting their utility for advanced applications such as machine learning, which rely on well-structured input to generate insights.

3.2.2 Emerging standards

To address these challenges, several standardized frameworks have emerged. The Allotrope Simple Model (ASM), developed by the Allotrope Foundation[22], is based on JavaScript Object Notation (JSON). Similarly, the Analytical Information Markup Language (AnIML), associated with the American Society of Testing and Materials (ASTM)[23], is built on Extensible Markup Language (XML).

In addition to these open standards, proprietary data schemas are also widely adopted in the industry. For example, Intermediary Data Schemas (IDS) by TetraScience[24] utilize a JSON-based approach for structuring laboratory data.

Despite these advances, the general adoption of these standards remains low. Laboratories still rely predominantly on flat files and distributed Excel sheets, maintaining the status quo [Della Corte *et al.* 2022; Steinwandter *et al.* 2019].

3.2.2.1 Allotrope

The ASM is a JSON-based standard for structuring instrument data, designed to be easily readable, writable, and transmittable by modern software systems. The data

[22] Allotrope Foundation, Washington, District of Columbia, USA

[23] ASTM International, West Conshohocken, Pennsylvania, USA

[24] TetraScience, Inc., Boston, Massachusetts, USA

structure is defined by JSON schemas, which are provided for a broad range of data types on the Allotrope GitLab repository [Allotrope Foundation 2023]. ASM uses key/-value pairs, where keys are defined by the Allotrope Foundation Ontology (AFO) to ensure consistency and standardization. This structure facilitates the capture of relevant experimental data and meta-data in a simple, human- and machine-readable format, promoting interoperability across devices and systems. Backed by a broad industry consortium, the standard is open and accessible, although not royalty-free for commercial use.

3.2.2.2 AnIML

AnIML is an ASTM XML standard for analytical chemistry and biological data. It comprises a core XML schema that defines how to store different kinds of analytical data, a technique schema for specific analytical techniques, and a set of technique definition documents that apply constraints to the flexible core. AnIML's design allows for extensions to accommodate vendor- and institution-specific data fields. Although royalty-free, due to intransparency regarding the specification, a lack of documentation, as well as no visible community or industry buy-in, this standardization effort has negligible impact.

3.3 Interface standardization

3.3.1 Challenges in laboratory automation and digitalization

Laboratories face significant barriers to achieving full digitalization. Fragmented solutions, often referred to as "island solutions" or "workcells", create data silos, requiring manual data transfer that increases inefficiencies and errors. For instance, a LIMS may not directly interface with a robotic liquid handler, necessitating custom-built integrations. Additionally, laboratories generate vast volumes of data, which are often locked in disparate laboratory software and data management systems, complicating efforts to implement FAIR data principles.

Another challenge is the lack of universal communication standards for laboratory instruments. While protocols like SiLA 2 and OPC UA have made strides, their adoption remains inconsistent. Regulatory requirements and guidance, such as compliance with Good Automated Manufacturing Practice (GAMP), further add to the complexity and cost of integrating diverse hard- and software systems. These challenges are particularly pronounced in laboratories reliant on legacy hard- and software, for which modernization often demands significant investment.

The core problem of the laboratory industry is the sheer endless amount of vertically[25], point-to-point integrated solutions, custom built for niche application areas. Confined to outdated computer operating systems, closed-off source code and a desolation of open interfaces for interoperable connectivity. The vast amount of hard- and software systems on the market, including the long tail of legacy laboratory instruments and desktop applications, poses an impossible task to any vertically integrated solution as the number of systems to connect is becoming to large. Most companies thus revert

[25] Vertical integration in software refers to the consolidation of multiple technology layers – such as the combination of a laboratory instrument, computer, operating system, application, and data storage - all within the solution scope of the same vendor.

to shallow integrations that do not integrate the actual hard- or software directly but bypass it via computer file system connectors. These shallow integrations can exchange Excel or comma-separated value (CSV) files with pipetting worklists or acquire measurement result files exported and stored as flat file the computer hard-drive, but do not enable a direct connection to the hard- and software system and are not suitable or sufficient for many digitalization and automation use cases.

3.3.2 Emerging standards

Industry-specific standards are pivotal in addressing interoperability challenges in laboratories. Establishing and pushing industry-wide standards can resolve some of these issues, but as responsibility and interface requirements remain unclear, adoption remains sluggish. The following standards are slowly gaining adoption by industry stakeholders, including pharmaceutical and life science companies, hard- and software vendors, as well as solution providers (often referred to as "Integrators"). In-depth comparisons of both standards can be found in the literature [Biermann *et al.* 2021; Freundel 2022; Gauglitz 2018; Habich *et al.* 2024; Hohmann 2021].

3.3.2.1 SiLA 2

The Standard in Laboratory Automation (SiLA) [SiLA 2 Consortium 2022; SiLA Consortium 2019] is an interface standard that is developed and funded by the SiLA Consortium[26] which consists of members active in the life science and pharmaceutical industry as well as academia. The latest release, SiLA 2 1.1 is fundamentally different to its predecessor SiLA 1 [Bär *et al.* 2012], which was initially released in 2009 and is now fully deprecated and not backwards-compatible. In order to ensure interoperability, SiLA 2 is more restrictive regarding the specification of the underlying technology and

[26]SiLA Consortium, Bubikon, Switzerland

the freedom that users have with the implementation.

SiLA 2 is a standardized communication protocol that uses Hypertext Transfer Protocol 2 (HTTP/2) based remote procedure call protocol (gRPC) and protocol buffers to define device interfaces using a client-server architecture [Juchli 2022]. The standard is openly accessible and royalty-free and reference implementations in multiple programming languages are maintained and funded by the consortium. The standard is gaining traction globally within the laboratory automation community with prominent supporters and users on the end-user side, including e.g. F. Hoffmann-La Roche[27], Unilever[28], and Novo Nordisk[29], as well as on the solution provider side, including e.g. Biosero[30] and Tecan[31] [Hinkel *et al.* 2023; Logan *et al.* 2024; Mione *et al.* 2024].

3.3.2.2 OPC UA LADS

The Open Platform Communication Unified Architecture, short OPC UA, is a standard maintained by the OPC Foundation[32], an industry consortium with members in the manufacturing industry and adjacent academia. The Laboratory and Analytical Device Standard (LADS), released in January 2024, is a companion specification of the OPC UA standard that extends the base standard with laboratory and domain specific information and specification. The development of the companion specification has been led by SPECTARIS e.V.[33], a German industry association of analytical laboratory equipment manufacturers [Brendel *et al.* 2022; OPC Foundation 2023] .

[27]F. Hoffmann-La Roche Ltd., Basel, Switzerland

[28]Unilever PLC, London, England, United Kingdom

[29]Novo Nordisk A/S, Bagsværd, Denmark

[30]Biosero Inc., San Diego, California, USA

[31]Tecan Group Ltd., Männedorf, Switzerland

[32]OPC Foundation, Scottsdale, Arizona, USA

[33]SPECTARIS e.V., Berlin, Germany

OPC UA is a communication protocol that is network protocol independent. Mappings to support a variety of network protocols, including Message Queuing Telemetry Transport (MQTT) and HTTP/1.1, have been developed by OPC Foundation members and the community. The standard is openly accessible and open-source reference implementations exist, but require royalty payments for closed-source commercialization.

Hildebrandt *et al.* [2020] have outlined the importance of interface-ontology mappings to provide valuable process and device information in automation systems in the manufacturing industry , which has been demonstrated by Steindl *et al.* [2021], by providing guidance on the translation of interfaces based on OPC UA to domain specific ontologies. Domínguez *et al.* [2022] provide additional information on this topic, including available literature, related resources, and implementation considerations .

3.4 Integration Solutions

Robotic instruments are central to modern laboratories, performing tasks such as liquid handling, sample preparation, and high-throughput screening. Examples include the Microlab STAR from Hamilton[34] and OT-2 from Opentrons[35]. While liquid handling vendors aim to support a large range of periphery devices with their software including incubators, centrifuges, and robotic arms, they typically support only a fraction or support only specific periphery device vendors, creating a pre-selection and additional lock-in for their customers.

For more complex systems that go beyond the capabilities of the liquid handling vendor software capability, scheduling software like Biosero's[36] Green Button Go Sched-

[34]Hamilton Bonaduz AG, Bonaduz, Switzerland

[35]Opentrons Labworks, Inc., Long Island City, New York, USA

[36]Biosero Inc., San Diego, California, USA

uler and CellarioScheduler from HighRes Biosolutions[37] step in. They connect to the respective vendor software on a high-level to ensure efficient operation of these robotic systems with a multitude of periphery devices, thus acting as an inbuilt, work-cell, and domain-specific integration platform.

Bioreactor systems are becoming increasingly complex as vendors incorporate industry needs for miniaturization, parallelization, and improved data generation and process control (see FDA Process Analytical Technology (PAT) initiative) into their products. However, bioreactor systems like the DASGIP 4x Parallel Bioreactor System[38], the Labfors bioreactor system[39], or the bioREACTOR 48[40] with their respective vendor software DASWare Control[41] and Infors Eve[42], are struggeling to provide comprehensive integration capabilites for enhanced data acquisition and control for the vast number of potential periphery and analytical instruments, software systems, and sensors that could be connected. This opens up a niche for yet another domain-specific breed of integration platforms such as Sm@rtLine Data Cockpit[43] or Lucullus[44]. These solutions serve as bioprocessing domain-specific integration platforms.

Integration solutions bridge the gap across vendor and instrument barriers. For example:

Scitara[45]: Focuses on creating unified data streams across systems by integrating disparate laboratory instruments and software. It enables real-time data transfer and

[37]HighRes Biosolutions Inc., Beverly, Massachusetts, USA

[38]DASGIP AG, part of Eppendorf AG, Hamburg, Germany

[39]Infors AG, Bottmingen, Switzerland

[40]2mag AG, Munich, Germany

[41]DASGIP AG, part of Eppendorf AG, Hamburg, Germany

[42]Infors AG, Bottmingen, Switzerland

[43]AGU Planungsgesellschaft mbH, Leverkusen, Germany

[44]Securecell AG, Urdorf, Switzerland

[45]Scitara Corporation, Marlborough, Massachussetts, USA

helps laboratories meet regulatory compliance standards by standardizing data formats. Although focusing on file system and software integration, they also integrate simple laboratory devices like balances and IoT devices for data extraction purposes.

Zontal[46], TetraScience[47], Ganymede[48]: Specialize in data acquisition, contextualization, and long-term storage, providing tools for digital archiving and retrieval. Their cloud-native data platforms standardize and harmonize raw scientific data from laboratory instruments into a consistent format. They focus on file system integration and the FAIRification of research data supporting either proprietary, but open schemas or the ASM.

ATU Sm@rtLine Data Cockpit[49]: Focuses on bioprocessing labs, offering real-time data acquisition and process monitoring. It integrates with various bioreactors, analytical and peripheral devices, providing visualization and analytics to support process control and bioprocess optimization.

Securecell Lucullus[50]: Designed for bioprocessing, Lucullus provides comprehensive process management by integrating data from sensors, bioreactors, and downstream processing units. Its capabilities include batch tracking, process visualization, and advanced control strategies for scaling and optimization. Integrations are bidirectional.

Integration solutions provide the connective layer that ensures data flows freely across systems. While these platforms address either a particular domain or data harmonization only, they lack generalizable and open automation capabilities, requiring additional software layers for complete lab digitalization and automation. This multi-layered ap-

[46]ZONTAL, Inc., Provo, Utah, USA

[47]TetraScience, Inc., Boston, Massachussetts, USA

[48]Ganymede, Inc., Palo Alto, California, USA

[49]AGU Planungsgesellschaft mbH, Leverkusen, Germany

[50]Securecell AG, Urdorf, Switzerland

proach adds complexity but is necessary for achieving interoperability in the current landscape.

3.5 Workflow standardization

3.5.1 Reproducibility and interoperability

The reproducibility crisis in science – a phenomenon where a substantial proportion of published research cannot be reliably replicated – is rooted in several systemic issues. Key among these is the absence of raw data, poorly defined experimental methodologies, and the lack of a common workflow description language [Merz *et al.* 2020; Miyakawa 2020]. Without standardized and openly executable workflows, researchers are forced to rely on vendor-specific, proprietary description languages for laboratory automation through vendor software. Examples of such software in the liquid handling domain are Tecan's FluentControl[51], Hamilton's VENUS[52], or Beckman's Biomek Software[53], which are neither interoperable nor transferable across different devices [Bartley *et al.* 2023]. This fragmentation creates inefficiencies, as scientists must learn and manage multiple languages for each device, and there is no guarantee that workflows written for one system can be adapted to another.

Proprietary languages also suffer from significant drawbacks in terms of version control and standardization. Changes to a protocol may not be easily tracked or documented, limiting transparency and traceability. Furthermore, these languages often do not integrate with a broader data ecosystem, leaving laboratories isolated and reliant on specific vendors. This isolation hampers progress toward creating truly interoperable,

[51] Tecan Group Ltd., Männedorf, Switzerland

[52] Hamilton Bonaduz AG, Bonaduz, Switzerland

[53] Beckman Coulter, Inc., Brea, California, USA

automated labs that can handle heterogeneous datasets and workflows in a standardized manner [Teytelman *et al.* 2016].

3.5.2 Emerging standards - An early picture

In response to these challenges, platforms like Protocols.io[54] have emerged. Protocols.io offers an open-access, semi-structured platform for developing, sharing, and versioning experimental protocols. While this approach represents a significant evolution from traditional text-based Standard Operating Procedures (SOPs), it is still grounded in manual, human-readable methodologies. Protocols.io enhances transparency and reproducibility by allowing step-by-step annotations, version control, and collaborative sharing among researchers, but it does not directly tie these protocols to specific devices or labware [Teytelman *et al.* 2016].

In contrast, Briefly Bio[55] takes a more structured approach, incorporating predefined elements into its protocol framework based on a bespoke laboratory process ontology. This allows for better organization and reuse of experimental workflows, moving one step closer to automation. However, like Protocols.io, Briefly Bio protocols remain non-executable as they lack integration with device-specific ontologies and cannot control or execute tasks on laboratory automation hardware [Bartley *et al.* 2023].

The gap between these semi-structured workflow description languages and fully automated laboratory workflows is being addressed by efforts such as LabOP[56]. LabOP aims to create a universal protocol description framework by building a device ontology that maps each instrument's capabilities to standardized features. While this approach represents significant progress toward enabling interoperability and automation, LabOP

[54]Springer Nature Group, Berlin, Germany

[55]Briefly Bio Ltd., London, England, United Kingdom

[56]Laboratory Open Protocol Language, open-source project, https://github.com/Bioprotocols/labop

protocols are still not directly executable on laboratory hardware. Instead, they serve as a bridge, translating protocol descriptions into a form that can be interpreted by more specialized automation tools [Bartley *et al.* 2023].

To achieve the vision of a truly connected and automated lab ecosystem, protocols must not only standardize workflows but also become directly executable on diverse automation platforms. This will require harmonizing different ontologies for data, devices, and workflows, as well as robust mappings between them. LabOP, although not a proclaimed standard for workflow description, but the closest to one there is, together with interface standards such as SiLA 2 and OPC UA offer promising frameworks for addressing these issues, but significant technical and cultural challenges remain in achieving widespread adoption. A unified effort to integrate structured, executable protocols with automation systems will be essential to overcoming the current barriers to reproducibility and efficiency in scientific research.

3.6 Economic consideration

Resource and time efficiency are a key priority in the life science industry, and digitalization promises vast potential. Even though a digital revolution is taking place, many pharmaceutical and biotechnology companies are lagging behind. They often rely on in-house software development, deployment, and maintenance. However, the employment of large IT departments and the cost of specialized software engineers is a huge cost factor that appears to counter-weigh the labor cost of skilled lab scientists and technicians [Scheitz *et al.* 2018] and thus digitalization and automation is stalled. Although the R&D spend on drug development in the pharmaceutical industry has increased 10-fold between 1984 and 2018 and the number of new drugs entering the market only doubling in the same period of time [US Congressional Budget Office 2021], building robust business cases for digitalization and automation solutions remains a challenge.

The literature on automation outlines mathematical approaches to calculate the economic feasibility of potential automation cases [Ravazzi *et al.* 2009]. While in manufacturing, the investment in terms of resources and the output – the product – are clearly defined, calculating a digitalization and automation case in life science R&D is not straight-forward due to the sheer number of potential and oftentimes unknown inputs and outputs. Understanding the effects of digitalizing and automating a process is important as they can also result in net negative outcomes [Ribeiro *et al.* 2023]. This leads to a decision-making process that is mostly based on tacit knowledge and that varies on a case by case basis.

4 The SiLA 2 Manager for rapid device integration and workflow automation

This publication lays the foundation of this thesis and other, following publications [Benner *et al.* 2023; Bromig *et al.* 2022b, 2023; Von den Eichen *et al.* 2021, 2022]. The presented software solution decouples the integration aspect of laboratory digitalization and automation from the application. While applications in biotechnology research are rapidly evolving and constantly changing, the instruments required have a vastly slower turnover and can be considered as part of the inventory of the laboratory.

As part of the work on the SiLA 2 Manager the hardware infrastructure of a typical laboratory for the microbial bioprocess design and analysis, was extended by a layer of software infrastructure comprising the development of a range of standardized, SiLA 2 compliant instrument connector applications and an integration platform software: the SiLA 2 Manager.

The SiLA 2 Manager is based on a microservice architecture that enables the connection to laboratory instruments across vendor barriers. These connections leverage the SiLA 2 standard which imposes a client-server architecture. The application consisting of a Python[1] backend and a TypeScript[2] frontend, offers a web-based interface secured by OAuth2[3] for user interaction. It supports real-time data collection and stor-

[1]Python Programming Language, Python Software Foundation, Wilmington, Delaware, United States

[2]TypeScript Programming Language, Microsoft, Redmond, Washington, United States

[3]Web Authorization Protocol, IETF Administration LLC, Wilmington, Delaware, United States

age in InfluxDB[4] databases, enhancing control and automation of laboratory processes. The framework is modular, based on industry standards, and open-source, promoting transparency and reducing third-party dependency. The development of custom automation applications is enabled by a workflow engine based on the Python programming language and Docker[5], a virtualization and containerization technology.

As part of the work towards this publication, a library of open source instrument connector applications has been developed. These connectors enable connectivity to bioreactor systems, peristaltic pumps, laboratory balances, off-gas analytics, optical density (OD) sensors, flowmeters and thermostats. Furthermore, the inclusion of SiLA 2 device drivers in a GitLab repository promotes open access and facilitates external control integration into automation workflows, supporting broader adoption and collaboration of standards in the field.

The biological use case investigates the impact of irregular intermittent substrate feeding on microbial fermentations during scale-up, specifically translating feeding dynamics observed at the mL-scale to the L-scale. The experimental setup includes a parallel bioreactor system[6] equipped with peristaltic pumps[7] and one off-gas analysis device per reactor (BlueVary, BlueSens gas sensor GmbH, Herten, Germany). Using the SiLA 2 Manager, feeding events derived from prior mL-scale bioreactor[8] experiments with *E. coli*, in which feeding was performed intermittently by a liquid handling station[9], are replicated on the L-scale to analyze their effect on metabolic responses such as dissolved oxygen levels. This use case was selected to demonstrate the software's cap-

[4]InfluxData Inc., San Francisco, California, United States

[5]Docker, Inc., Palo Alto, California, United States

[6]DASGIP Parallel Bioreactor System, DASGIP AG, part of Eppendorf AG, Hamburg, Germany

[7]Masterflex Ismatec Reglo ICC, Avantor, Inc., Pennsylvania, United States

[8]bioREACTOR 48, 2mag AG, Munich, Germany

[9]Microlab STARlet, Hamilton Bonaduz AG, Bonaduz, Switzerland

ability to integrate heterogeneous laboratory hardware, automate complex workflows, and enable reproducible, cross-scale bioprocess experimentation through standardized device control and data acquisition.

For this publication, I was primarily responsible for the conceptualization, design, and development of the software system. I developed the entire software stack, including all individual SiLA 2-compliant connector applications, as well as the overarching microservice architecture of the SiLA 2 manager. The software architecture and implementation strategy were conceptualized independently by me. I also contributed substantially to the validation, hardware testing, methodology, and data curation. In addition to writing the original draft of the manuscript and creating the visualizations, I oversaw the overall project execution and coordination.

SoftwareX 17 (2022) 100991

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

The SiLA 2 Manager for rapid device integration and workflow automation

Lukas Bromig, David Leiter, Alexandru-Virgil Mardale, Nikolas von den Eichen, Emmeran Bieringer, Dirk Weuster-Botz *

Institute of Biochemical Engineering, Technical University of Munich, Boltzmannstr. 15, 85748 Garching, Germany

ARTICLE INFO

Article history:
Received 9 March 2021
Received in revised form 23 August 2021
Accepted 11 January 2022

Keywords:
Laboratory automation
Device integration
SiLA2
Data integrity
Biotechnology
Life sciences
Software framework
Web application
IoT

ABSTRACT

Rapid integration of laboratory devices into automated workflows remains an arduous and time-consuming process. A lack of interface standardization among hardware vendors leads to inflexible device setups and highly customized and expensive software solutions. Thus, practical integration capabilities of existing laboratory control systems (LCS) are insufficient when it comes to developing small, rapid and cost-efficient automation solutions. The increasing importance of data integrity as well as software system and workflow documentation, due to cGMP regulations, is calling for structured, reliable and transparent middleware solutions. The SiLA 2 Manager uses the emerging SiLA 2 standard and provides a lean and extendable framework for device discovery, management, and workflow design, thereby bridging the gap between the physical device and higher software levels. This Internet of Things (IoT) application enables immediate control of SiLA 2 devices and ensures user-friendly real-time data collection and storage.

Code metadata

Current code version	v1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-21-00051
Code Ocean compute capsule	*n/a*
Legal Code License	MIT
Code versioning system used	Git
Software code languages, tools, and services used	Python, Typescript, Shell, Bash, HTML, Angular, FastAPI, Docker, SiLA 2, PostgreSQL, Redis, InfluxDB, Nginx
Compilation requirements, operating environments & dependencies	Linux, Unix-like, Microsoft Windows, Python $\geq$ 3.7, node.js $\geq$ 12
If available Link to developer documentation/manual	https://sila2-manager.readthedocs.io/en/latest/
Support email for questions	lukas.bromig@tum.de

Software metadata

Current software version	v.1.0
Permanent link to executables of this version	https://gitlab.com/lukas.bromig/sila2_manager
Legal Software License	MIT
Computing platforms/Operating Systems	Linux, Unix-like, Microsoft Windows
Installation requirements & dependencies	See installation documentation and requirements.txt
If available, link to user manual	https://sila2-manager.readthedocs.io/en/latest/
Support email for questions	lukas.bromig@tum.de

* Corresponding author.
E-mail address: dirk.weuster-botz@tum.de (D. Weuster-Botz).

https://doi.org/10.1016/j.softx.2022.100991

1. Motivation and significance

Laboratory automation in the commercial and academic environment in the life and chemical sciences has been gaining momentum over the past two decades [1,2]. The trend of process parallelization and miniaturization in the chemical and biochemical laboratory, coupled with Design of Experiment (DoE) based software assisted experiment planning is calling for increasingly automated experimental setups and advanced capability for digital device interaction [3–6]. The main driving force is not only increased efficiency and throughput, but also automated documentation and a reduction in human error. Regulatory bodies have expanded their guidelines to keep up with technological advances in laboratory automation, encompassing not only audit trail documentation, but documentation and validation of software and computerized systems as well (cGMP, EudraLex). Increasingly detailed documentation requirements can only be fulfilled efficiently by digitized systems [7–9].

Background and problem statement

Computerized systems in a laboratory environment are hierarchical constructs of several subsystems. They range from the enterprise or laboratory-level information management systems (LIMS) to the laboratory or process control software down to the so-called middleware on the device level. In most cases, the process control software directly integrates the necessary devices without the use of a designated middleware. Prominent examples are liquid handling stations that operate with several periphery devices and require dedicated software for their operation. Access to the respective periphery devices is only possible through the vendor software interface. The integration of new devices is difficult and, due to the closed-source nature of the software, requires application specialists. Furthermore, such software solutions generally lack an appropriate application programming interface (API) for further integration with other process control software or into higher-level organizational systems such as LIMS.

A variety of vendor-specific, proprietary software and hardware interfaces exist for standalone laboratory equipment. Custom solutions that combine several devices to create automated workflows have long development times and require programming and software engineering expertise. This makes the integration of individual devices into an automated workflow an arduous task. The combination of various standalone devices with each other, like a digital scale with a robotic arm or the integration of an unsupported third-party device into a liquid handling station (LHS), enables a more efficient mode of operation. However, the lack of standardized device interfaces and of appropriate middleware still impedes the development of automated workflows [10,11].

Hence, larger automation systems grow in software and hardware complexity and require subject matter experts. Replacement or upgrading of individual devices is not straightforward and the most often used closed-source, non-modular software solutions lead to low code reusability and a high long-term maintenance cost [12,13]. Due to a lack in hardware interface standardization, most commercial automation solutions focus on workflow and audit trail documentation or data and user management, but not basic middleware for hardware integration.

Despite recent advances in high throughput system technology and a constantly broadening automation product range, laboratory automation is still most often confined to custom-built island solutions [2,7]. These solutions, whether implemented as closed or open systems, are highly inflexible and come at a high development cost. The resulting cost to automate a task in contrast to the cost of the respective manual labor, i.e. a tradeoff between the amount of repetitions versus the complexity of the task, still defines the grade of automation in the laboratory environment to this day. This tradeoff is an old paradigm which may hold true for high-throughput stations but falls short when considering the laboratory environment as a whole. There are many aspects to consider in laboratory automation, such as gained flexibility, replaceability, integration into existing systems, time savings and thus competitive advantage, minimization of human error, reduction of sample volume and data integrity [12,14,15]. Reducing development cost through the use of innovative software for device integration directly will make lab automation more affordable and result in the aforementioned benefits.

Laboratory digitization can be overwhelming and the amount of combinations of architectural solutions for distributed computing (REST, SOAP, JSON-/XML-RPC, gRPC), programming languages and database types is vast. Developing custom solutions that integrate all kinds of devices is possible and has been shown by Porr et al. (2020) [11]. However, such solutions are quickly increasing in complexity and maintenance effort making rapid integration difficult. Laboratory automation should not be the problem, but the solution. Reducing the aforementioned complexity, in accordance with the law of parsimony, should be the main priority of any automation solution. Converging on a minimal number of network protocols, programming languages, databases and overall software dependencies is a key aim of this software proposal.

Standardization in laboratory automation - SiLA 2

Device integration requires intricate knowledge about the used communication protocol. From a user perspective, this turns simple commands like "Start_pump" into a programming challenge of e.g. string parsing, checksum calculations, numeral system conversions and worries about thread safety or buffer queue size. Furthermore, the intuitive assumption that the integration solution of one device is transferable to another device with the same functionality, but from different vendors, fails immediately if the first glance at the documentation reveals that two entirely different communication protocols are used, effectively doubling the development effort. In an ideal world, the end-user should not have to worry about such integration complexities and a workflow script written for a specific task should only care about the device type and be independent of the device make. However, temporal and structural behavior of existing interfaces is vastly different and error messages and error handling capabilities are generally incompatible with third-party products. Introducing a standardized device interface as an additional level of abstraction resolves these problems and reduces the users effort to a simple "Start_pump"-command, resulting in rapid integration capability and device interchangeability [16].

Application of a standardized interface, as proposed by the not-for-profit membership organization SiLA (SiLA, Rapperswil-Jona, Switzerland), can drastically reduce the complexity and development times of automation solutions. SiLA 2 provides the tools for rapid integration and enables device-agnostic comprehensive middleware platforms for a new generation of laboratory devices with plug-and-play capability. According to the standard, each device is implemented as a SiLA server that exposes its service in the network as a set of features. Each feature comprises a set of functionally related commands and properties available for the client to call. A SiLA device is a special case of a SiLA server, as a SiLA server can implement any kind of service. Network communication is based on gRPC via HTTP/2. The client can be incorporated in a higher-level software, such as in a LIMS or a PCS, or as in the case of the proposed middleware software, a simple workflow script [14,17–19].

The number of embedded SiLA 2 compatible devices is increasing steadily, but adoption is still in its infancy as hardware vendors have not yet fully committed resources to in-house implementations. However, since SiLA is a software solution to a hardware interface problem, any legacy device can be converted into a SiLA 2 device in a fast and cost-effective manner as shown by Porr et al. (2020) [20]. This makes every piece of laboratory hardware a potential addition to an automation workflow. Moving forward, a SiLA server that implements a laboratory device will be referred to as a service. There are current efforts to specify a standard based on OPC-UA, which is widely used in the closely related process industry. The OPC-UA Laboratory Agnostic Device Standard (LADS) may compete with SiLA 2 in the future [21].

Expectations and comparison with existing tools

A modern process control software framework should be modular and based on industry standards. The used technology must be widely adopted to ensure long-term support and, as the software itself, be open-source to ensure transparency and reduce third-party company dependency. Industrial adoption of open-source software – often held back by unjustified quality concerns – is increasing, especially if value-added services are offered [22]. Prominent examples of successful open-source software are Linux or the databases MySQL and InfluxDB.

The dogma of limiting device access to software installed on a local computer is outdated. Devices should be accessible by the local network in a device-as-a-service fashion. Henceforth, the controlling software should be readily accessible from any eligible end user device. A further major advantage of this web-based concept is the resulting operating system or platform independence.

In the age of information, all data is important and data recording should start at the lowest level: the device level. Raw data and meta-data should be collected not just in regulated environments, but in academia and research as well [15]. Gathering comprehensive datasets is time-consuming and prone to human error. Thus, any laboratory software should be capable of linking certain streams of data to a selected database which can be connected to the laboratory or company LIMS.

The SiLA Browser (UniteLabs AG, Basel, Switzerland) is a free-to-use closed-source IoT application that enables discovery and direct control of SiLA 2 devices. However, it is unsuitable for workflow automation due to the lack of a scripting environment or workflow editor. Other notable closed-source software solutions are niceLab (EQUIcon Software GmbH, Jena, Germany) and zenLab® (Infoteam Software AG, Bubenreuth, Germany), the successor of the iLab [23], whereas the zenLab®, a scalable middleware automation framework, follows the similar core principles as the proposed software, but accompanied by the restrictions of proprietary vendor software. The SiLA 2 Manager provides a free, open-source alternative for researchers, engineers, educators, and small laboratories in general, to automate their workflows. In comparison to the SiLA Browser, the proposed software is a functional open-source alternative and vast expansion.

Workforce change management in the laboratory environment in regard to new software solutions is an often underestimated barrier due to a low level of computer and programming skills. This further raises the importance of standardized, plug-and-play like device interfaces and intuitive graphical user interfaces. Intuitive, browser-based solutions that offer easily accessible device-as-a-service functionality are needed to transform the laboratory environment into an automated workspace [24].

2. Software description

2.1. Software architecture

The SiLA 2 Manager is based on a client–server architecture, with a python backend and a typescript frontend. The user interacts with the software through the web-based frontend which communicates with the backend API by HTTP and WebSockets. The frontend is secured by using the open authorization protocol OAuth2. Authentication of authorized users is managed by issuing a JSON Web Token. The automatic token timeout and renewal policy can be configured to meet the required security level. Fig. 1 shows the relationship between the user and the involved software systems. A user interacts with the application by using the web-based graphical interface to discover and control available services or execute experimental scripts. The SiLA 2 Manager relies on multicast DNS service discovery (zeroconf) and is able to discover SiLA servers that multicast their service by default. A generic SiLA 2 python client uses the provided information to connect to any of the discovered services. Once connected, the user can explore the devices features, execute commands and integrate the devices in experimental workflow scripts.

Furthermore, SiLA 2 services can be linked to InfluxDB databases (v.1.7) [25] to automatically store selected process information for the duration of an experiment. This stored service data, e.g. from a device, can be accessed from within the scripting environment with the respective python database client. Stored information can also be accessed and visualized via the web-based database browser chronograf, which is part of the influxdata stack.

To allow further expansion and customization, the frontend and backend have been strictly separated. The python backend uses the FastApi web framework to transfer data via HTTP and WebSocket to and from the Angular frontend. Fig. 2 shows the different components of the system and the interactions between them. Each component is a container, e.g. a module, that can be executed independently.

To store persistent application data, e.g. service or booking information, a relational SQL database is used because of its high reliability, guaranteed data consistency and ease of use. In addition to this, the relational data model, with its ability to efficiently join rows from different tables, makes it a natural fit for the persistent application data of the SiLA 2 Manager. The open-source database PostgreSQL (v.13) was selected because it is fast and has excellent SQL conformance [26]. Most No-SQL database alternatives benefit from high horizontal scalability, i.e. they can distribute data over many servers and load balance the access to it. However, this does not provide any real benefit for the proposed use case since there are no large distributed datasets. The disadvantage of making it more difficult to ensure precise data consistency would outweigh the advantages provided by the horizontal scalability of NoSQL database types. Further, the ability to have very loosely defined non-structured data structures is of no benefit and thus not a relevant decision criterion for the presented application.

The application consists of several subprocesses as shown in Fig. 2. The in-memory database Redis (v.6.0.9) is used as a message broker to enable interprocess communication. An in-memory, No-SQL database was chosen over an SQL database because information transmission between 2 components of the system rather than persistent storage is required. Redis is a well-established and high-performance database [27]. A major advantage, as compared to other messaging libraries, is that Redis can be used for other complex operations, like fast in-memory storage or caching, as well.

The user is provided with an Angular client for easy access to the functionalities of the system, which are exposed over an HTTP

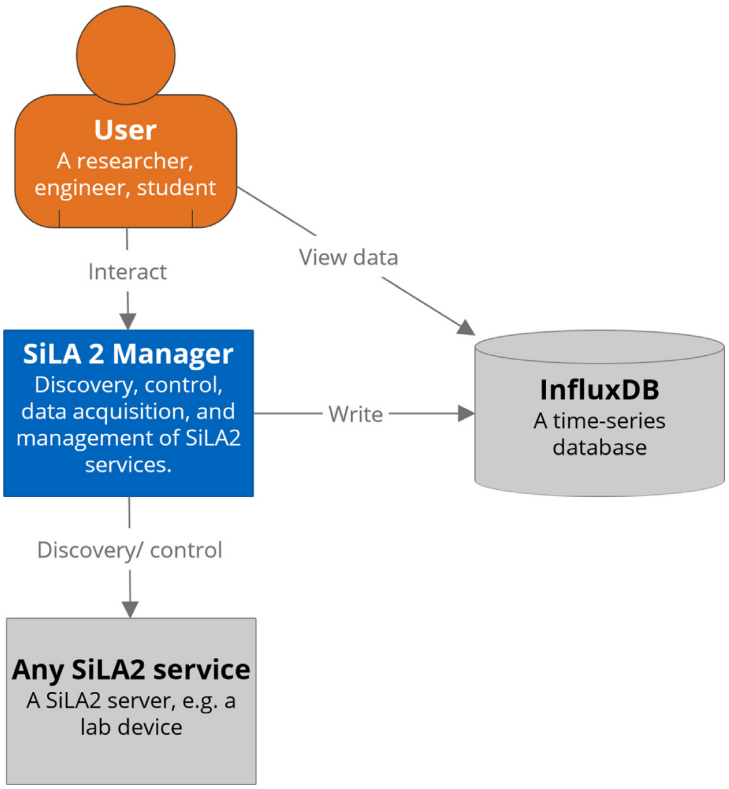

Fig. 1. The context diagram showing the scope of the software and its relation to the user and other software systems. The user interacts with the software through the web-frontend and may view process data through web-frontend of the database. The SiLA 2 Manager discovers new services in the network via the mDNS multicast server registrations of the SiLA servers and connects to the services via gRPC using a generic SiLA client. Recorded process data is forwarded from the SiLA 2 Manager to a selected InfluxDB database.

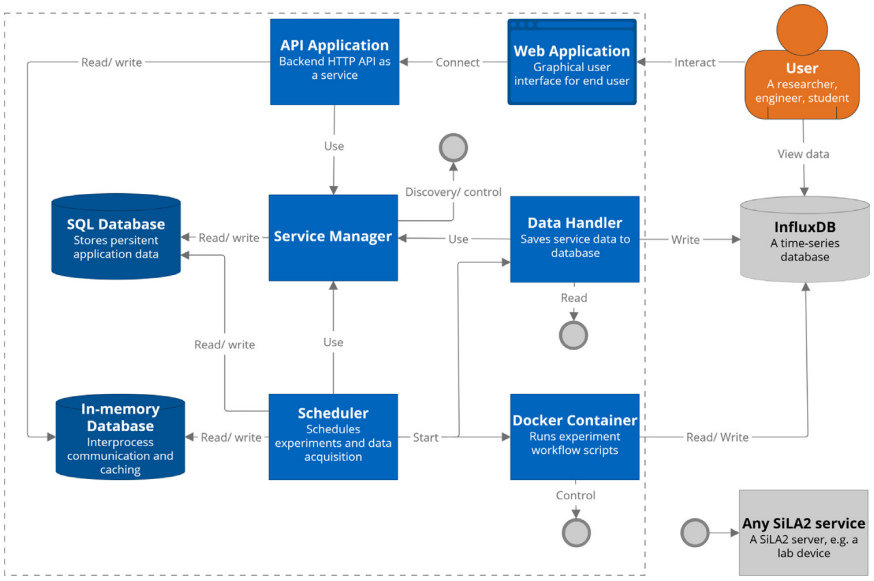

Fig. 2. The container diagram of the SiLA 2 Manager is a detailed description of the software system introduced in Fig. 1. The web-frontend makes API calls to the backend API. The API forwards these calls to the main sub-system, the service manager. A PostgreSQL database is used to store application related data. Temporary and quickly changing data storage, such as device status and log data, is handled by the in-memory database Redis by using publish and subscribe (PUBSUB) channels. The scheduler is an independent service that controls experiments in a docker container and initiates data transfer via the data handler.

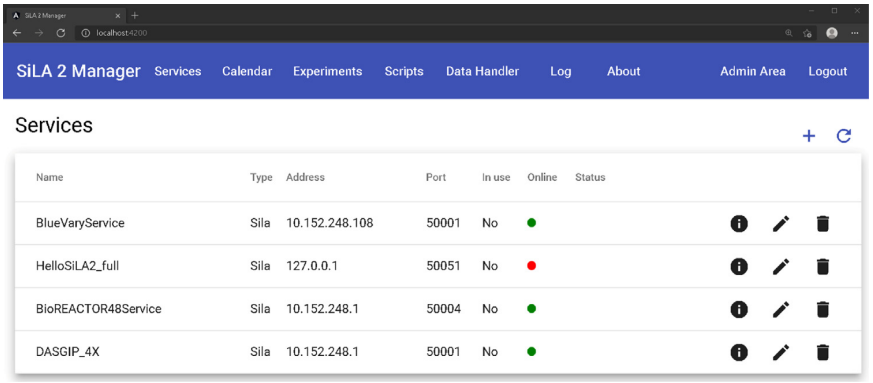

Fig. 3. The homepage shows registered services and the top layer of the service tree that contains more detailed information on the service's features, commands and properties.

Rest-like API through the FastAPI Python framework. The service manager handles detection of services on the network, communication with services, and storage and retrieval of persistent information (e.g. service, experiment, and time-series database connection details) to and from the PostgreSQL database. The angular frontend is served by nginx [28], which is also used as a reverse proxy for the FastApi backend. Angular was chosen because it is a proven technology to create modern single page web applications.

The scheduler is an independent subprocess that is responsible for the execution of experiments and the initialization of the data handler and docker containers. The scheduler periodically checks for new experiment booking entries in the PostgreSQL database and queues it in its in-memory scheduling list. With the provided buttons to start and stop experiments, the user can interact with the scheduler directly over a WebSocket connection to the backend, which itself communicates with the standalone scheduling process over the in-memory database Redis. When an experiment is ready to be scheduled, the scheduler loads the corresponding data and script from the PostgreSQL database and launches a docker container and an associated data handling process. The user-provided script contains a predefined entry point, which gets called with a dictionary of instantiated device objects and their name as parameter.

A docker container is used to execute a Python script. SiLA services can be incorporated into this user script to control the laboratory devices. Experiment associated script and service data are loaded from the SQL-database and copied into the docker container as a tar archive file at the beginning of an experiment. Docker allows the execution of python scripts in isolation; therefore, the scripts cannot interfere with each other or compromise the host system. Containers can also be run independently of the host operating system. This is crucial to guarantee server security and stability. The default python docker image (Python 3.8.3-alpine) is used as it is an appropriate base image for most use cases and is based on the lean Alpine Linux distribution. At the same time, the data handler is started and periodically saves user-specified data from the services into an InfluxDB time-series database. Polling intervals and the selection of data to be stored are configured in the data-handler.

A detailed documentation of the software architecture, the installation process and several application examples are available in the software documentation in the Git-repository or online on readthedocs.io (https://sila2-manager.readthedocs.io/en/latest/).

2.2. Software functionalities

The homepage for the SiLA 2 Manager software is shown in Fig. 3 and can be accessed, upon login, by entering the IP of the host computer followed by the default port in the URL-address bar of any modern browser, e.g. 127.0.0.1:4200 for a local host. There are six main tabs, each dedicated to one of the core functionalities as well as an about page with instructions and additional information. The core components are: Services, Calendar, Experiments, Scripts, Data Handler, and Log. User management and user authentication is handled in the tabs Admin Area and Login/Logout. The main page is the Services tab in which all registered services and respective important information is shown. New services are discovered and added in this view and the expanded card of the services will open the more detailed service tree view and the interactive command execution environment.

Service manager and browser - Services

The core functionality of the proposed software is service discovery, control, and management. All of these are available on the main page. SiLA services can be added either by using the discovery mode or by manual specification of the server address and IP. Previously added SiLA services are listed as shown in Fig. 3. When adding a new service, the service discovery initiates a scan of the local network and returns the registered SiLA servers within. SiLA services are registered on the unicast domain name system (DNS) server and can be identified by their hostname which contains '*sila.local*' as terminal identifier. Services are assigned the server name but can be renamed upon addition. Furthermore, the service is assigned an internal universally unique identifier (UUID) and the service details are stored in the PostreSQL database. A dynamic SiLA 2 python client is used to establish a connection with the service and access information regarding implemented features and functions. According to the SiLA standard, the features of a SiLA service are defined in the feature definition language (FDL), which is based on the extended markup language XML. Several FDL-files describe the full functionality of the service. During the initial connection to the SiLA 2 server, these files are queried by calling standard functions of the SiLAService feature, a feature that is always implemented by default. All further communication with the SiLA 2 server is realized by the dynamic client which is part of the SiLA 2 python library. Other implementations, such as C# and the C++, also offer a generic (dynamic) client as part of the distribution.

Fig. 4. The bottom layer of the service tree allows direct execution of command and property calls. If no parameter is supplied for command calls, the default parameter will be used. in case of device services, accidental execution may have serious unwanted consequences; thus, execution must be confirmed.

The new device will appear in the service list and, if a connection is established successfully, will be shown as online. Clicking on the service name or the information icon, the service detail tree is expanded as shown in Fig. 4. The service tree shows all implemented features of the service and the functions they comprise. According to the SiLA specifications, a SiLA service must implement features containing functionally related commands and properties, which can be observable or unobservable. Standard features are implemented by default, whereas custom features are device specific. Each feature can be further expanded to show command and property call information. While service properties are static (e.g. a device serial number) or dynamic values (e.g. device time or status), commands require a parameter to be supplied with the command call. Equipment that requires special security considerations can be further secured by mandatory incorporation of the SiLA Standard Features "AuthenticationFeature" and "LockFeature" in the SiLA Server of that device, making command execution only possible with valid access credentials on unlocked devices.

Scripting environment - Scripts

The scripting environment enables the user to write automation scripts that incorporate the registered services as shown in Fig. 5. In contrast to device specific vendor software, the scripting environment's functionality is non-restrictive. Scripts can be uploaded, created, and edited with the embedded python editor. Each user only has access to their own scripts to avoid unwanted access or script execution by unqualified users. The editor is based on the open-source monaco code editor and supports type hinting, auto-completion, and syntax control for python-based scripts. Devices are automatically instantiated, and commands and properties can be called using the SiLA python syntax of the generic client. For ease of use, the respective usage syntax is explicitly stated in the service detail view of each call. External data-sources and non-standard python packages can be imported but must be specified in the dockerfile.

Workflow creation - Experiments

Processes can be executed by scheduling experiments. An experiment consists of an experiment name, start and end date, a workflow script, and a selection of services. New experiments can be added which will open the experiment definition window (Fig. 6). In this window, the desired script to be executed, as well as the required services are selected. The start and expected end-time of the process are specified. The used services will be reserved and blocked for the requested time slot. At the time of execution, the specified script is run in a docker container and the services are marked as "currently unavailable" in the service list overview. Automated experiment workflows require access to device services, a script that orchestrates device operation, and a database to store relevant information. The experiments view enables the user to setup such workflows and schedule their execution. In a commercial setting, each part of the workflow, i.e. the SiLA 2 servers of the devices and the workflow script, would need to be validated and access to these workflows be restricted by using the inbuilt user management and authentication system.

The main view of the experiments tab shows a list of scheduled experiments, accompanied by the most important information such as the start and end time of the experiment, booked services, the user script and the current status of the experiment. Scheduled experiments automatically create a booking entry in the calendar view. It is possible to start experiments before their scheduled starting time by pressing the play icon. A running experiment can be aborted prematurely. An embedded terminal displays the output of the docker container to observe the current status of the script execution.

Bookings and calendar overview - Calendar

If a service is assigned to a specific experiment, a booking is automatically created for the entire timeframe of the experiment. A service can only be assigned to an experiment if it is available throughout the experiments start and end time. The calendar page visualizes these bookings and provides the user with additional information on the booking, such as the respective experiment, script and the user who created it. Furthermore, it is possible to create bookings manually. In a future release, shared or part-time service access will be available for bookings as well. It is possible to delete bookings manually. Even automatically created bookings can be deleted. However, this would circumvent the in-built security mechanism and may lead to services being accessed by multiple experimental scripts at the same time, as the script does not communicate with the booking system anymore once started. This may be desirable if the SiLA service implements a virtual device or some other software solution, such as a DoE or data analysis software.

Database connection - Data handler

Data collection is enabled by default, although the user should specify which data is collected. This configuration is done on the data handler page shown in Fig. 7. InfluxDB databases can

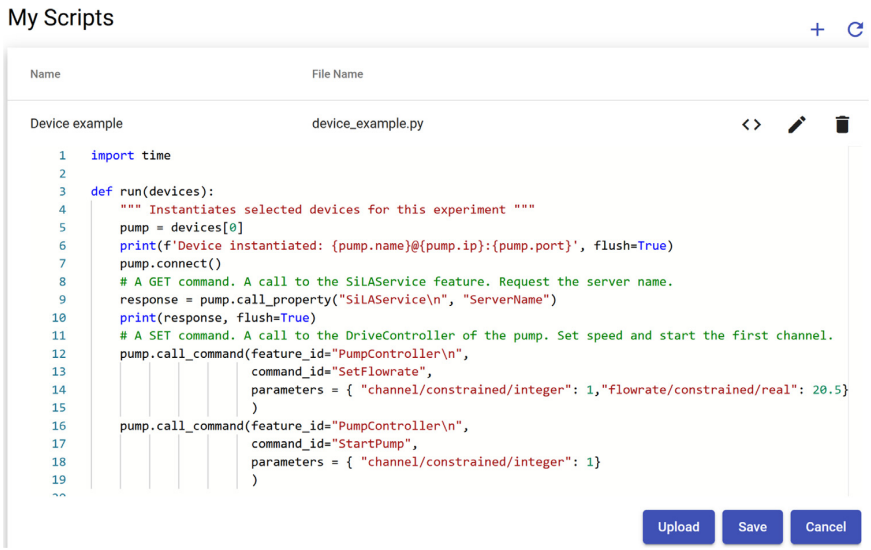

Fig. 5. An interactive code editor is used to view, create and edit workflow scripts. Scripts can be uploaded and saved. The device services are automatically instantiated and passed to the *run()* function as attribute. They are accessed either by key index in the order of assignment (see experiment overview for order) or by key name, which is identical to the name the device services are listed as in the service list overview. In this example code snippet, a peristaltic pump is imported. A get and a set command is executed to demonstrate the interface syntax. The command syntax is shown in the detailed service list view (See Fig. 4) for each command to further simplify the process.

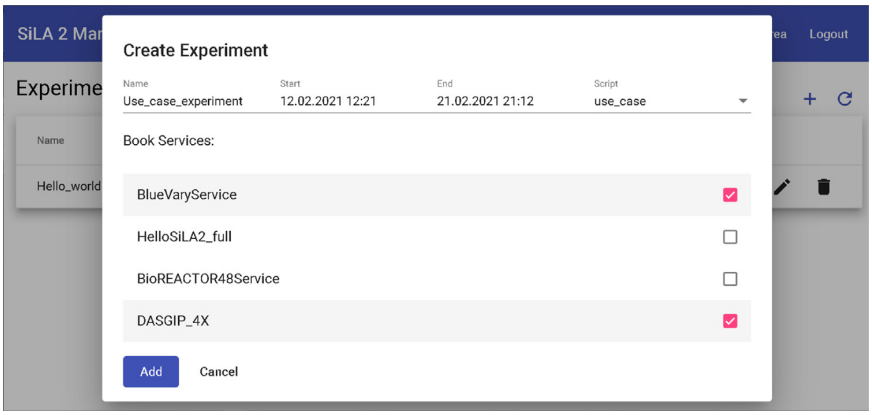

Fig. 6. An example experiment is defined by selecting a workflow script, the used services, and by specifying the start time of execution and the anticipated end time. The logging output of the script is forwarded to the frontend and can be viewed in a terminal when the experiment is expanded.

be registered and linked to services. InfluxDB is a time-series database that is well suited for experimental time-series data [29, 30]. To use this feature, an InfluxDB server must be running within the network. Providing the connection details to the SiLA 2 Manager is sufficient. A username and password can be added optionally for additional security. A registered database can be linked to a service to setup automated data transfer by clicking the link symbol. Data transfer is started when the booking of a service commences, i.e. the experiment the service is used in, is started.

The data handler will repeatedly execute the SiLA calls in the user-specified polling intervals and store the responses in the linked database with experiment name, service name, and user-name as tags. Several options are available to configure the data calls, the data type, and the polling interval. The SiLA 2 Manager is service agnostic and the functional response of individual calls is unknown. Therefore, the user must specify the calls that should be executed periodically and must deactivate unwanted calls, such as certain set-commands. If set-commands are to be called, a parameter can be supplied.

Most types of data can be classified as either meta-data or measurement data. Typically, meta-data does not need to be queried frequently. In most cases, requesting meta data (device ID, calibration data, etc.) hourly or just once at the beginning of an experiment is sufficient. Measurement data (Temperature, pressure, etc.) on the contrary is usually queried on a more frequent basis. The data handler distinguishes between the two data types. Since there is no way to distinguish the type of data queried by a call automatically in a reliable fashion, the user can specify the type for each command using the meta-checkbox. Depending on the selection, a default value is implemented (1 h for meta-data, 30 s for measurement data). Different users have different needs regarding polling intervals. Therefore, the defaults can be overwritten to transfer data according to a custom polling interval.

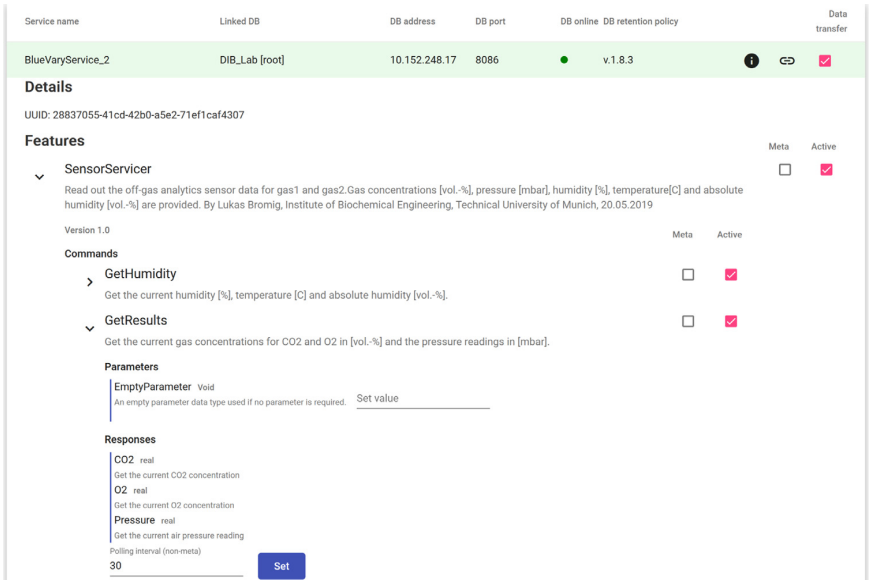

Fig. 7. The service tree for the data handler configuration. Data transfer can be de-/activated for each property and command. The user assigns a meta-data or measurement data flag which determines the polling interval used. If no custom polling interval is supplied, the default polling intervals are used. Some command calls, like most set commands, may require additional parameters. The example above shows a configured off-gas analysis device that is linked to an InfluxDB. Sensor results related to gas concentrations, pressure, and humidity are polled in the default non-meta data interval of 30 s.

The data handler configuration is stored in the PostgreSQL database. The lower level of the data handler service tree is shown in Fig. 7. A customized configuration is crucial to disable the undesired execution of set commands or the storage of unnecessary data.

The data handler simplifies data-acquisition and encourages collection of all data and meta-data for improved data integrity. The separation of the data acquisition from the user script used in the experiment has several advantages:

1. The query calls are not part of the user-script, improving readability and making the script shorter.
2. Reduces the amount of code that needs to be written by the operator.
3. Data-acquisition is out-sourced to a separate process. This way data-acquisition is guaranteed to continue in case an experiment crashes.
4. The data can be easily accessed from within the user-script by a respective python client library. An example script is provided in the "Scripts"-section of the application.

2.3. Example and application use cases

Example use cases

Use cases can stretch from very simple routine tasks to highly complex, event-driven process control workflows. A simple use case is scheduled device calibration (e.g. a digital scales) or self-maintenance/cleaning tasks, or the collection of basic sensory laboratory data such as room temperature and pressure. Setting up the latter would be the simplest use case. Add the sensor device to the SiLA 2 Manager, customize the data you want to collect, link the device to a database, schedule an experiment with an empty script and start the experiment. The data-handler will now collect the data and meta-data of your device and store it in your database.

Application use case

The impact of intermittent substrate feeding on microbial fermentation processes is a key factor for successful scale-up in bioprocess development and remains an active area of research [31,32]. Frequently changing nutrient availability leads to rapidly changing metabolic states and may result in population heterogeneity [33]. The effect of intermittent substrate feeding on microbial process performance can be studied in laboratory-scale parallel reactor systems to mimic the quasi-intermittent feeding in large scale reactors due to nutrient gradients caused by non-ideal mixing.

Using a parallelized and miniaturized bioreactor system with 48 single-use stirred-tank reactors on a mL-scale (BioREACTOR48, 2mag AG, Munich, Germany) automated by a LHS (Hamilton STARLet, Hamilton Bonaduz AG, Bonaduz, Switzerland) enables a fast analysis of the process parameter space and thus the quantification of the effect that the substrate feeding strategy has on product formation and population heterogeneity. In this application example, a scale up experiment is performed for a selection of substrate feeding intervals, to ensure that the observed influence on microbial protein expression is the effect of the varying feeding interval length and that this effect is independent of reactor size. To achieve this, the substrate feeding strategy must be emulated as closely as possible on the L-scale. Substrate addition on the mL-scale is realized by pulse addition with the pipetting robot. A dynamic scheduling algorithm coordinates this feeding event with other tasks, such as sampling, at-line measurements or pH-control, resulting in small variations of the feed interval duration. The challenge at hand is the direct translation of these irregular events into pump action on the L-scale.

The experimental setup consists of a parallel stirred tank bioreactor system with four bioreactors (DASGIP Parallel Bioreactor System, Eppendorf, Hamburg, Germany) that is connected to a proprietary control computer and an off-gas analysis device. The vendor software DASware® control (DASware® control, Eppendorf, Hamburg, Germany), with its integrated OPC-UA interface, serves as access point for the SiLA server of the DASGIP bioreactor

system. Peristaltic pumps used for substrate addition, pH-control, and addition of antifoam are periphery devices of the reactor system. A gateway module, based on a BeagleBone Green microcomputer and provided by Porr et al. (2020) [20], is connected to the off-gas analysis device by RS-232. The corresponding SiLA server is hosted by the microcomputer and uses the serial interface for device control. Both SiLA servers are based on the *sila2lib* release 0.2.5 of the open-source SiLA python repository and the code was generated using the python *sila2codegenerator* version 0.2.0.

The vendor software of the L-scale reactor system offers only limited scripting capability. Scheduling of non-trivial substrate addition events, especially if based on external files, is not possible. Furthermore, the used off-gas analytics are from a different vendor (BlueVary, BlueSens, Herten, Germany). Although the off-gas concentrations are not used actively for process control in this particular use case, it is desirable to store and monitor all generated data in and from a single source, i.e. the laboratory database (InfluxDB, Influxdata, San Francisco, USA). In this setup the SiLA Manager is used for data and meta-data acquisition of both devices as well as the advanced orchestration of the feed pump events, whereas the simple tasks of pH-, DO-, and temperature control are carried out by DASware® control. Establishing full control with the SiLA 2 Manager is possible but was not realized to keep the application example as simple as possible.

The resulting workflow is written in form of a python script and is uploaded to the SiLA 2 Manager. DASware® control is started and device connection established and verified. At the scheduled experiment start time, the docker container is created and the user script is executed automatically. Substrate additions performed by the liquid handling station during the 48 preceding experiments on a mL-scale are stored in the same database that is used for this experiment. Thus, the timestamps and the added substrate volumes of the discrete feeding events can easily be extracted from the database using the *influxdb-client* within the user script. For reasons of simplicity and reproducibility, a different approach was chosen to avail the required data within the docker environment. Experimental data regarding the substrate feeding events and volumes during the experiment in the BioREACTOR48 were extracted from the database in a csv-file format and manually moved to the data folder of the docker environment directory. The csv-file is provided as part of the use-case example in the Git repository. Files in this directory are automatically moved into the docker container. This feature enables the user access to any kind of file from within the scripting environment.

Any logic necessary to control substrate addition events on the L-scale is based on the provided data and is defined in the python user script a priori. Data acquisition for both devices, the reactor system and the off-gas analysis device, is started automatically according to the selected criteria in the data handler settings. Hence, no additional commands related to data acquisition need to be specified in the workflow script. Workflow progress is monitored through the embedded terminal on the experiments page and measurement data can be visualized through the browser-based database interface chronograf. All process related data is stored with a respective experiment and user tag to enable quick and fast data selection. Further workflow script examples dealing with device and database integration as well as simplified version of the use case described above can be found in the example folder of the project Git repository.

Fig. 8 shows the resulting dissolved oxygen concentration in the fermentation with *E. coli* during a short window of the cultivation and compares the immediate catabolic response to a glucose feeding event on the mL- and the L-scale. It can be seen that the irregular glucose feeding events could be synchronized with each other which enables a direct comparison of the metabolic

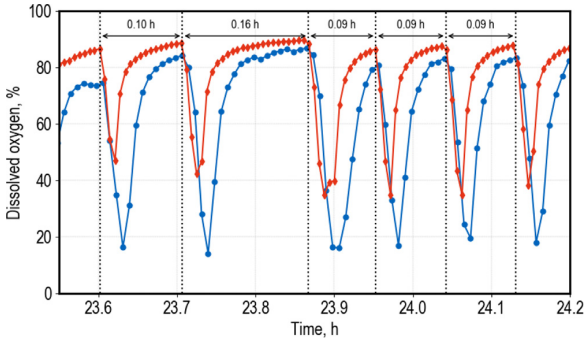

Fig. 8. The effect of intermittent glucose feed events on the dissolved oxygen level is shown for cultivations of *E. coli* in the mL- and L-scale. Feed events (▪▪▪) are unevenly distributed on the mL-scale (●) due to the dynamic nature of the task scheduling algorithm of the pipetting robot. The feeding pattern is mimicked as closely as possible with a peristaltic pump on the L-scale (♦) to investigate scale-up limitations/criteria. The resulting DO off-set is caused by different oxygen transfer rates. Stirrer speeds of the mL- and L-scale were set at 3000 rpm and 1100 rpm respectively.

response, i.e. the dissolved oxygen concentration. The remaining offset (minimum DO and maximum DO) can be attributed to different oxygen transfer rates of the stirred-tank reactors at mL- and L-scale and the different DO sensors. Dry cell mass concentrations were similar on both scales within the estimation error (14.3 ± 0.4 gL^{-1} on the mL- and 15.8 ± 0.9 gL^{-1} on the L-scale at 24.3 h cultivation time).

The presented use case only involves two devices. A more sophisticated example, including more devices, event-based actions, and active use of the influx data for process control would have been beyond the scope of this article. However, the presented example clearly demonstrates the advantages of simple integration and automated, centralized data acquisition.

3. Impact

The proposed software enables automation and data acquisition of common tasks and complex scientific experiments. Based on the SiLA 2 standard, it reduces integration effort of laboratory devices. Its intuitive GUI design and expandable software framework reduces the necessary experience to automate experimental setups in the long run. Device integration is decoupled from developing automation workflows. Furthermore, it is a beneficial development tool for testing of self-written SiLA servers. This software is published under the MIT license to allow for continuous application in science and industry. Addressing a common laboratory challenge results in a broad application spectrum beyond disciplines in biotechnology, the life and chemical sciences.

4. Conclusions and future development

The proposed software is mainly, but not exclusively, aimed at laboratories of the life sciences, biochemical sciences and chemical sciences working in research and development, that are planning to digitize their laboratory infrastructure and create automated, SiLA-based, workflows including devices from multiple vendors. Programming-agnostic users can use the basic SiLA-browser functionality, with its service discovery and direct control capability, while the workflow creation requires basic scripting experience in the python programming language. The powerful scripting environment allows the rapid integration of laboratory devices and databases into automated workflows and

empowers the user with the full spectrum of python packages. The data handler component enables the simple setup of database connections and encourages enhanced acquisition of measurement and meta-data according to the FAIR data principles: Findability, accessibility, interoperability, and reusability [34].

User management, device and script-specific access rights are critical when dealing with potentially dangerous or complicated equipment to avoid erroneous or malicious use. The proposed software incorporates these basic features that will be extended in future efforts to improve the overall operational safety of this software. Whereas device specific safety features, such as emergency shutdown procedures or preconditions for device start-up, should be handled by the respective SiLA Server of that device, the orchestration of multiple devices in a complex workflow requires intricate knowledge of the used resources.

The proposed software offers an implementation framework – a model – that bridges the gap between standardized device interfaces, such as SiLA 2, and higher-level control software. By clearly separating the challenges of device integration from customized workflow automation solutions, a holistic approach is chosen. By providing integrated devices as a network service, this software can be broadly applied and reduces development times.

It is a helpful tool for developers and researchers alike and can also be used as basis for a more comprehensive laboratory control software. Future work on this project will include an expansion of the experiment planning component, to enable advanced interaction capabilities with the running docker container, and event-based chaining of several individual experiments. A multi-level structure of main and sub-routines is planned for the scripting environment to increase code reusability. It is planned to include a scheduling algorithm to allow the sharing of resources during the runtime of multiple, simultaneously active workflows. SiLA 2 implementations are being actively developed and are quickly progressing towards meeting the full standard specifications, hence this software will be maintained to ensure long-term compatibility.

CRediT authorship contribution statement

Lukas Bromig: Conceptualization, Software, Validation, Methodology, Data curation, Writing – original draft, Visualization, Supervision, Project administration. **David Leiter:** Software, Validation, Methodology, Data curation, Writing – review & editing. **Alexandru-Virgil Mardale:** Software, Validation, Methodology, Data curation, Writing – review & editing. **Nikolas von den Eichen:** Investigation, Formal analysis, Data curation, Writing – review & editing, Visualization. **Emmeran Bieringer:** Investigation. **Dirk Weuster-Botz:** Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors thank the German Ministry of Education and Research (BMBF) for funding within the national joint research project Digitalization in Industrial Biotechnology (DigInBio). Grant number: FKZ 031B0463B.

The authors thank Dr. M. M. Porr, Institute of Technical Chemistry, University of Hannover, for providing the gateway modules used for the integration of legacy devices.

Lukas Bromig and Nikolas von den Eichen thank for the support provided by the TUM Graduate School (Technical University of Munich, Germany).

References

[1] Neubauer P, Glauche F, Cruz-Bournazou MN. Editorial: Bioprocess development in the era of digitalization. Eng Life Sci 2017;17(11):1140–1. http://dx.doi.org/10.1002/elsc.201770113.

[2] Chapman T. Lab automation and robotics: Automation on the move. Nature 2003;421(6923):661–3. http://dx.doi.org/10.1038/421661a, Number: 6923 Publisher: Nature Publishing Group.

[3] Haby B, Hans S, Anane E, Sawatzki A, Krausch N, Neubauer P, et al. Integrated robotic mini bioreactor platform for automated, parallel microbial cultivation with online data handling and process control. SLAS TECHNOL: Transl Life Sci Innov 2019;24(6):569–82. http://dx.doi.org/10.1177/2472630319860775, Publisher: SAGE Publications Inc.

[4] Selekman JA, Qiu J, Tran K, Stevens J, Rosso V, Simmons E, et al. High-throughput automation in chemical process development. Annu Rev Chem Biomol Eng 2017;8:525–47. http://dx.doi.org/10.1146/annurev-chembioeng-060816-101411, Publisher: Annual Reviews.

[5] McMullen JP, Jensen KF. Integrated microreactors for reaction automation: New approaches to reaction development. Annu Rev Anal Chem 2010;3(1):19–42. http://dx.doi.org/10.1146/annurev.anchem.111808.073718.

[6] Puskeiler R, Kusterer A, John GT, Weuster-Botz D. Miniature bioreactors for automated high-throughput bioprocess design (HTBD): reproducibility of parallel fed-batch cultivations with escherichia coli. Biotechnol Appl Biochem 2005;42(3):227–35. http://dx.doi.org/10.1042/BA20040197.

[7] Bliesner DM. Laboratory control system operations in A GMP environment. John Wiley & Sons; 2020.

[8] USFDA. Part 211 - Current good manufacturing practice for finished pharmaceuticals (21CFR211). In: Code of federal regulations. 2019, URL https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/CFRSearch.cfm?CFRPart=211.

[9] EudraLex. Annex 11: Computerised systems. In: Good manufacturing practice - medicinal products for human and veterinary use - the rules governing medicinal products in The European. Union Brussels, SANCO/C8/AM/SI/Ares(2010)1064599. 2011, URL https://ec.europa.eu/health/documents/eudralex/vol-4_en.

[10] Hawker CD. Laboratory automation: total and subtotal. Clin Lab Med 2007;27(4):749–70. http://dx.doi.org/10.1016/j.cll.2007.07.010, Publisher: Elsevier.

[11] Porr M, Lange F, Marquard D, Niemeyer L, Lindner P, Scheper T, et al. Implementing a digital infrastructure for the lab using a central laboratory server and the SiLA2 communication standard. Eng Life Sci 2020;n/a(n/a). http://dx.doi.org/10.1002/elsc.202000053.

[12] Fleischer H, Thurow K. Automation solutions for analytical measurements: Concepts and applications. John Wiley & Sons; 2017.

[13] Ampatzoglou A, Kritikos A, Kakarontzas G, Stamelos I. An empirical investigation on the reusability of design patterns and software packages. J Syst Softw 2011;84(12):2265–83. http://dx.doi.org/10.1016/j.jss.2011.06.047.

[14] Wolf A, Galambos P, Széll K. Device integration concepts in laboratory automation. In: 2020 IEEE 24th International Conference On Intelligent Engineering Systems (INES). 2020, p. 171–8. http://dx.doi.org/10.1109/INES49302.2020.9147171.

[15] Gauglitz G. Lab 4.0: Sila or OPC UA. Anal Bioanal Chem 2018;410(21):5093–4. http://dx.doi.org/10.1007/s00216-018-1192-6.

[16] Bär H, Hochstrasser R, Papenfuß B. SiLA: Basic standards for rapid integration in laboratory automation. J Clin Lab Autom 2012;17(2):86–95. http://dx.doi.org/10.1177/2211068211424550.

[17] Sila 2 working group, SiLA2 base repository. GitLab 2018. last visited: 08.03.2021. URL https://gitlab.com/SiLA2/sila_base.

[18] Sila 2 part (a) - overview, concepts and core specification. last visited: 08.03.2021. URL https://sila2.gitlab.io/sila_base/.

[19] SiLA. SiLA standard | SiLA rapid integration. SiLA-Standard 2018. last visited: 08.03.2021. URL https://sila-standard.com.

[20] Porr M, Schwarz S, Lange F, Niemeyer L, Hentrop T, Marquard D, et al. Bringing IoT to the lab: SiLA2 and open-source-powered gateway module for integrating legacy devices into the digital laboratory. HardwareX 2020;8:e00118. http://dx.doi.org/10.1016/j.ohx.2020.e00118.

[21] Ladwig B. Geräteintegration entlang des labor-workflows: Ein neuer standard für das smarte labor. 2020, p. 9.

[22] Nagy D, Yassin AM, Bhattacherjee A. Organizational adoption of open source software: barriers and remedies. Commun. ACM 2010;53(3):148–51. http://dx.doi.org/10.1145/1666420.1666457.

[23] Schmid I, Aschoff J. A scalable software framework for data integration in bioprocess development. Eng Life Sci 2017;17(11):1159–65. http://dx.doi.org/10.1002/elsc.201600008.

[24] Ng IC, Wakenshaw SY. The Internet-of-Things: Review and research directions. Int J Res Mark 2017;34(1):3–21. http://dx.doi.org/10.1016/j.ijresmar.2016.11.003.

[25] Shahid J. InfluxDB documentation. Release; 2019, URL https://buildmedia.readthedocs.org/media/pdf/influxdb-python/latest/influxdb-python.pdf.

[26] Makris A, Tserpes K, Spiliopoulos G, Zissis D, Anagnostopoulos D. Mongodb vs postgresql: A comparative study on performance aspects. GeoInformatica 2020. http://dx.doi.org/10.1007/s10707-020-00407-w.
[27] Jing H, Haihong E, Guan L, Jian D. Survey on nosql database. In: 2011 6th international conference on pervasive computing and applications. 2011, p. 363–6. http://dx.doi.org/10.1109/ICPCA.2011.6106531.
[28] Reese W. Nginx: The high-performance web server and reverse proxy. Linux J 2008;2008(173):2:2, URL https://dl.acm.org/doi/abs/10.5555/1412202.1412204.
[29] Nasar M, Kausar MA. Suitability of influxdb database for iot applications. Int J Innov Technol Explor Eng 2019;8(10):1850–7. http://dx.doi.org/10.35940/ijitee.J9225.0881019.
[30] Giacobbe M, Chaouch C, Scarpa M, Puliafito A. An implementation of influxdb for monitoring and analytics in distributed IoT environments. In: International conference on the sciences of electronics, technologies of information and telecommunications. Springer International Publishing; 2018, p. 155–62. http://dx.doi.org/10.1007/978-3-030-21005-2_15.
[31] Heins A-L, Reyelt J, Schmidt M, Kranz H, Weuster-Botz D. Development and characterization of escherichia coli triple reporter strains for investigation of population heterogeneity in bioprocesses. Microb Cell Factories 2020;19(1):14. http://dx.doi.org/10.1186/s12934-020-1283-x.
[32] Heins A-L, Weuster-Botz D. Population heterogeneity in microbial bioprocesses: origin, analysis, mechanisms, and future perspectives. Bioprocess Biosyst Eng 2018;41(7):889–916. http://dx.doi.org/10.1007/s00449-018-1922-3.
[33] Vasilakou E, van Loosdrecht MCM, Wahl SA. Escherichia coli metabolism under short-term repetitive substrate dynamics: adaptation and trade-offs. Microb Cell Factories 2020;19(1):116. http://dx.doi.org/10.1186/s12934-020-01379-0.
[34] Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, et al. (2016) The FAIR guiding principles for scientific data management and stewardship. sci data, 3, 160018. 2016, http://dx.doi.org/10.1038/sdata.2016.18.

5 Control of Parallelized Bioreactors I: Dynamic Scheduling Software for Efficient Bioprocess Management in High-Throughput Systems

Bioprocess development and optimization are essential and time-consuming steps along the biotechnology R&D value chain that can be greatly accelerated by the miniaturization and parallelization of bioreactors and experimental setups. Scaling down benchtop reactor experiments to the mL-scale and scaling up the number of reactors to 48 in parallel, while maintaining process quality, scalability, and reproducibility, is a challenge. To achieve this, bioreactor systems can be integrated into liquid handling stations, which also provides additional at-line assay capabilities. However, enabling process control of such a system is a challenge as systems from multiple vendors must be integrated, and custom application logic must be developed. Traditional vendor software is inflexible for such use cases.

This publication presents a novel dynamic scheduling software aimed at accelerating bioprocess development and optimization using a mL-scale parallel bioreactor system that is integrated into a high-throughput liquid handling station. The developed scheduling application addresses the limitations of conventional liquid handling system software, which are constrained by fixed sequential task execution, often leading to bottlenecks and inadequate process control when dealing with biological systems.

The software integrates with the liquid handling station[1], the 48 parallel bioreactor system[2], sensory[3] and analytical equipment[4] to optimize process control dynamically based on feedback from the biological system. By employing data-driven real-time prioritization and user-defined constraints, the software improves process control and resource allocation, thus enhancing the scalability and reproducibility of bioprocesses on the mL-scale.

The software application resulting from this work enables a variety of use cases including high-throughput protein expression studies in microbial systems [Blums *et al.* 2025; Von den Eichen *et al.* 2022] and the cultivation of microalgae [Benner 2024; Benner *et al.* 2023].

In this publication, I conceptualized the software architecture and was responsible for the majority of the software development and testing. My contributions include the implementation of the dynamic prioritization algorithm, which is central to the orchestration of the liquid handler for control of parallelized bioreactors, as well as the development of the SiLA 2-compliant connector applications required for instrument integration. While certain software components were developed by co-authors, I led the overall system design and implementation. I also validated the software, created all figures, and wrote the original draft of the manuscript. All authors contributed to the critical revision and final approval of the manuscript.

[1]Microlab STARlet, Hamilton Bonaduz AG, Bonaduz, Switzerland

[2]bioREACTOR 48, 2mag AG, Munich, Germany

[3]Multi-sensor bar for DO and pH, PreSens GmbH, Regensburg, Germany

[4]Biotek Synergy H1, Agilent Technologies, Inc., Santa Clara, California, United States

Bioprocess and Biosystems Engineering (2022) 45:1927–1937
https://doi.org/10.1007/s00449-022-02798-6

RESEARCH PAPER

Control of parallelized bioreactors I: dynamic scheduling software for efficient bioprocess management in high-throughput systems

Lukas Bromig[1] · **Nikolas von den Eichen**[1] · **Dirk Weuster-Botz**[1]

Received: 20 June 2022 / Accepted: 3 October 2022 / Published online: 18 October 2022

Abstract
The shift towards high-throughput technologies and automation in research and development in industrial biotechnology is highlighting the need for increased automation competence and specialized software solutions. Within bioprocess development, the trends towards miniaturization and parallelization of bioreactor systems rely on full automation and digital process control. Thus, mL-scale, parallel bioreactor systems require integration into liquid handling stations to perform a range of tasks stretching from substrate addition to automated sampling and sample analysis. To orchestrate these tasks, the authors propose a scheduling software to fully leverage the advantages of a state-of-the-art liquid handling station (LHS) and to enable improved process control and resource allocation. Fixed sequential order execution, the norm in LHS software, results in imperfect timing of essential operations like feeding or Ph control and execution intervals thereof, that are unknown a priori. However, the duration and control of, e.g., the feeding task and their frequency are of great importance for bioprocess control and the design of experiments. Hence, a software solution is presented that allows the orchestration of the respective operations through dynamic scheduling by external LHS control. With the proposed scheduling software, it is possible to define a dynamic process control strategy based on data-driven real-time prioritization and transparent, user-defined constraints. Drivers for a commercial 48 parallel bioreactor system and the related sensor equipment were developed using the SiLA 2 standard greatly simplifying the integration effort. Furthermore, this paper describes the experimental hardware and software setup required for the application use case presented in the second part.

Keywords Parallel bioreactors · Dynamic scheduling · High-throughput systems · Automation · Device integration

Introduction

New challenges arise with the advances in high-throughput technologies and the parallelization and miniaturization of bioreactor systems. Miniaturization is the means to achieve high-throughput bioreactor systems by parallelizing mL-scale bioreactors in liquid handling stations [1–6]. The miniaturization of bioprocesses supports the early stages of development by providing fast and cost-effective solutions for scale-up and process optimization studies [6]. The authors have shown in previous work that these systems provide a powerful toolset for bioprocess optimization by conducting fast protein expression studies [4]. However, deviations in process behavior due to the change of scale must be minimized and accounted for to ensure scalability of the experimental results to L- or pilot-scale reactors [7–10]. A good overview of current parallel bioreactor technology is given by Achinas et al. and Junne et al. [11–13].

Automated bioreactor control in liquid handling stations poses several challenges, as there are multiple parallel tasks that need to be performed to provide the same process conditions compared to the L-scale, such as pH-control and substrate addition. Alternate approaches directly translate the techniques from the L-scale to the miniaturized mL-scale by incorporating microfluidic systems into small, often disposable parallel reactor systems [14]. This leads to a labor-intensive and expensive solution that does not leverage the flexibility and power of liquid handling stations [11, 15].

By integrating the bioreactor system on the deck of a liquid handling station (LHS), these tasks can be handled by the pipetting station, which also greatly increases the types of operations that can be performed during the fermentation such as sample preparation and analysis [16]. However,

✉ Dirk Weuster-Botz
dirk.weuster-botz@tum.de

[1] Chair of Biochemical Engineering, Technical University of Munich, Boltzmannstraße 15, 85748 Garching, Germany

this strategy can lead to differences regarding scalability in comparison to conventional L-scale benchtop bioreactors, because the execution of multiple tasks and their execution frequency is limited by the availability of the LHS pipetting channels. This results in a bottleneck problem that requires a scheduling solution.

Using an LHS for substrate addition limits the possible feeding strategies to intermittent feeding [4, 8, 17, 18]. As a result of the aforementioned bottleneck the minimum time interval at which feeding operations can be performed is restricted. This lower bound is defined by the speed of the liquid handling station, the number of parallel operations it can perform, the deck layout of the setup (i.e., distance to travel) and the number of tasks to be executed simultaneously. Other technical constraints of the LHS include the pipetting volume and modes, as well as the number of parallel channels. While many of those can be improved by upgrading the hardware or by a change of deck layout, a major contribution towards the mitigation of this problem depends on the execution order of the required tasks and the optimization thereof especially in bottleneck situations in which sampling and feeding tasks overlap.

Introducing dynamic scheduling algorithms into the proprietary software of a LHS is a cumbersome task, as the software has not been designed to handle such specific use cases. Regular implementations usually consists of fixed events or repeated, sequential execution of tasks. Thus, a dedicated software is required that is either implemented directly in the LHS software or implemented as an external software that controls the LHS. The software fedbatchXP (DASGIP, an Eppendorf SE company, Jülich) was aimed to solve bioprocess control with dynamic task scheduling and has been employed in past publications [1, 2, 7, 15, 19–21]. However, the software has been discontinued. Among others, the main drawbacks of this software were its proprietary, compiled nature, which made it impossible to freely customize and extend the software, as well as the intransparent nature of the prioritization algorithm. Furthermore, the software was designed for a specific device setup, a bioREACTOR48 integrated in a Tecan Freedom Evo, and was not open for integration of more or other devices, or further future development.

External control of liquid handling stations is gaining importance as laboratory infrastructure is becoming more digitized and processes more sophisticated. The authors propose a data-driven, dynamic scheduling solution with external LHS control. The software re-evaluates priorities of user-defined tasks in real time and orchestrates their execution for bioprocess control of a miniaturized and parallelized bioreactor system.

Use cases for the proposed dynamic scheduling software are manifold and software solutions that simplify the complexity of automated miniaturized bioprocess systems are required not just in industry, but academia as well [22]. Due to the complex nature of biological experiments, it is difficult to predict the changes in pH or the effect of different feeding strategies in advance. Conventional LHS software is designed to build fixed, sequential process workflows. However, the knowledge required to design a suitable execution sequence is not available a priori and rather the outcome of successful process development. Furthermore, sequential execution can lead to severe problems when tasks cannot be performed on time. With a flexible execution order, such tasks are performed on demand and not just on schedule.

An example area is general process optimization and the research on population heterogeneity in the scale-up of fermentation processes [23–25]. Deviations from ideal reactor behavior, such as changing levels of dissolved oxygen due to varying substrate availability resulting from intermittent feeding, have been linked to population heterogeneity [26, 27]. Moving between scales, non-ideality may vary in kind and magnitude, resulting in differences in product, metabolite, or inhibitor concentration or process variables like optimal harvesting time to achieve high space–time yield [28]. The proposed scheduling software enables researchers to further investigate and quantify these deviations by setting defined feeding intervals. If sequential execution was applied, the interval would be undefined a priori. Furthermore, the investigation of multiple intervals during a single run is enabled, while maintaining good process control of pH and the acquisition of frequent samples. Hence, the proposed software can speed up the screening process on a small scale.

Dynamic scheduling can improve process control as it remains flexible in regard to the execution order, but at the same time retains a stable and defined average execution interval. By re-evaluating sensor data and recalculating priorities of the individual tasks, an improved resource allocation can be achieved. The authors propose a software solution that consist of multiple separate services: a dynamic scheduler (LHS Scheduler), a broker that enables external control over the Liquid handling station (LHS Server), a digital twin (LHS Simulator) that can be used to emulate the LHS for *in silico* tests, as well as SiLA 2 device drivers for the bioreactor system and the related sensor equipment.

Hardware and integration

The presented solution relies on the integration of multiple devices from different vendors. To overcome the barriers of varying proprietary device interfaces the standard *Standardization in Laboratory Automation* (SiLA 2) is used [29–31]. The SiLA standard is based on a server–client architecture and uses the remote procedure call protocol gRPC for communication, which is based on the standards HTTP/2 and

protocol buffers. The developed software interfaces, i.e., SiLA Servers, for the parallel bioreactor system and the pH/DO sensors are built according to this standard.

Parallel stirred-tank bioreactor system

The 48 × parallel stirred-tank bioreactor system (bioREACTOR 48, 2mag AG, Munich, Germany) is used for microbial fermentations on the mL-scale. The parallel bioreactor system was integrated into the software environment using the SiLA 2 standard. A SiLA Server was developed to translate the proprietary serial communication protocol (RS-232) of the hardware to provide the digital interface in the local network environment. The scheduler software controls the parallel bioreactor system with this SiLA 2 interface by a SiLA 2 client. Available functions include the starting and stopping of the reactor agitation, changing of the applied power, as well as the rotational speed settings. Furthermore, the interface can be used to obtain status data of the in-built Hall sensors, enabling the scheduler software to monitor and react to individual stirrer malfunctions. A simulation mode providing realistic data on, e.g., stirrer malfunctions was integrated to allow for *in silico* testing of the scheduler software. The SiLA 2 Server of the bioREACTOR48 was created using the SiLA 2 Python reference implementation [32] and is available as open-source project [33].

pH/DO sensor bars

Dissolved oxygen (DO) and pH of the 48 bioreactors is measured in parallel by 6 fluorometric reader bars with 8 sensors each (MCR, PreSens GmbH, Regensburg, Germany), which are placed in a compartment underneath the mL-scale bioreactor vessels [5]. The developed PreSens SiLA Server standardizes the proprietary serial communication protocol and allows automated data acquisition of process parameters by the scheduler software via the local network. The PreSens SiLA 2 Server includes a simulation mode, which provides mock data for testing purposes. The SiLA 2 Server of the PreSens sensor bars was implemented using the SiLA 2 Python reference implementation [32] and is available as open-source project [33].

Liquid handling station

A liquid handling station (Microlab® STARlet, Hamilton Bonaduz AG, Bonaduz, Switzerland) with eight 1000 µL pipetting channels is used to perform tasks on the 48 × parallel bioreactor system for the automated cultivation of microorganisms. The LHS is equipped with a microtiter plate (MTP) reader (Synergy HTX, BioTek, Winooski, USA) and a MTP washer (405 LS, BioTek, Winooski, USA), which are directly integrated into the hardware setup as periphery devices accessible by the LHS plate handler tool (iSWAP®, Hamitlon Bonaduz AG, Bonaduz, Switzerland).

A software package supplied by the vendor includes a method editor and a runtime environment (Microlab® STAR Software VENUS version 4.5, Hamilton Bonaduz AG, Bonaduz, Switzerland). Within the method editor, a user-friendly environment with method libraries and drivers for periphery devices is provided, which were used to integrate the MTP washer and reader. However, more complex methods or foreign device integrations require the use of the underlying vendor-specific Hamilton Standard Language (HSL) or custom support. Due to the number of external devices to be integrated and the complexity of a dynamic prioritization algorithm, a solution written in HSL was not an option and a method had to be found to take control of the LHS externally through the scheduling software. The device was integrated using the COM-Interop interface of the vendor software and a newly developed gRPC server. The gRPC server acts as a broker and enables external control of the LHS via the local network. A more detailed account of the external control is explained in the LHS Server section.

Example application setup

The target applications of the LHS Scheduler are microbial cultivations in the miniaturized bioreactor system BioREACTOR48 which is integrated into a liquid handling station. Figure 1 shows the setup for which the software has been developed and tested.

Software architecture and implementation

General software architecture

The software solution consists of multiple services, which are connected to the central scheduler application as shown in Fig. 2. The reactor system and the pH/DO sensors are connected to the LHS Scheduler by a standardized SiLA 2 server–client connection, whereas the integration of the LHS is realized through a gRPC broker server, which evaluates, processes, and forwards traffic between the scheduler application and the LHS. Furthermore, the scheduler application is connected to two databases to persist process and application data.

InfluxDB

InfluxDB (InfluxDB v.1.7.11, InfluxData, San Francisco, USA) is an open-source NoSQL time-series database [34, 35]. The scheduler application uses InfluxDB to store the acquired process data that is received from the external devices in real time. This includes data of the bioreactor

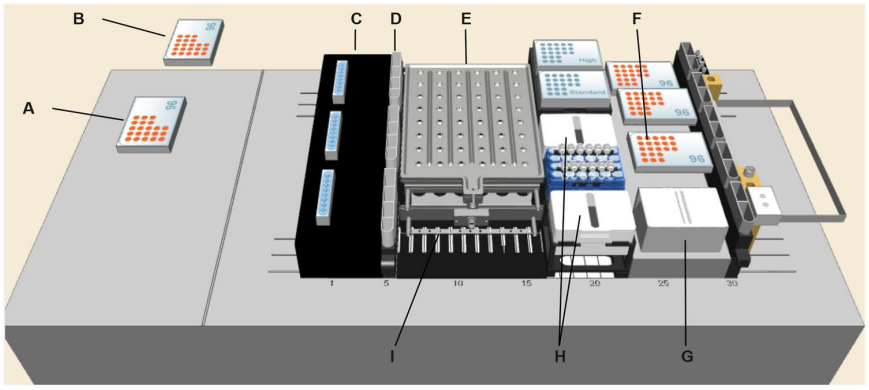

Fig. 1 A 3D visualization of the LHS deck layout showing the accessible entities and their positions as shown by the control software VENUS Run Control. From left to right, the setup consists of a MTP washer (**A**), a MTP reader (**B**), a pipetting needle wash station (**C**), several containers with aqueous EtOH (70% v/v) for pipetting needle disinfection (**D**), a bioREACTOR48 (**E**), a MTP for sample de-aeration, dilution and preparation (**F**), a container for substrate storage (**G**), containers for pH-controlling (NaOH) and phosphate buffered saline (**H**), and the socket of the pH/DO sensor bars (**I**, bars not shown). The plate reader and plate washer are accessed by the integrated plate gripper (iSWAP™)

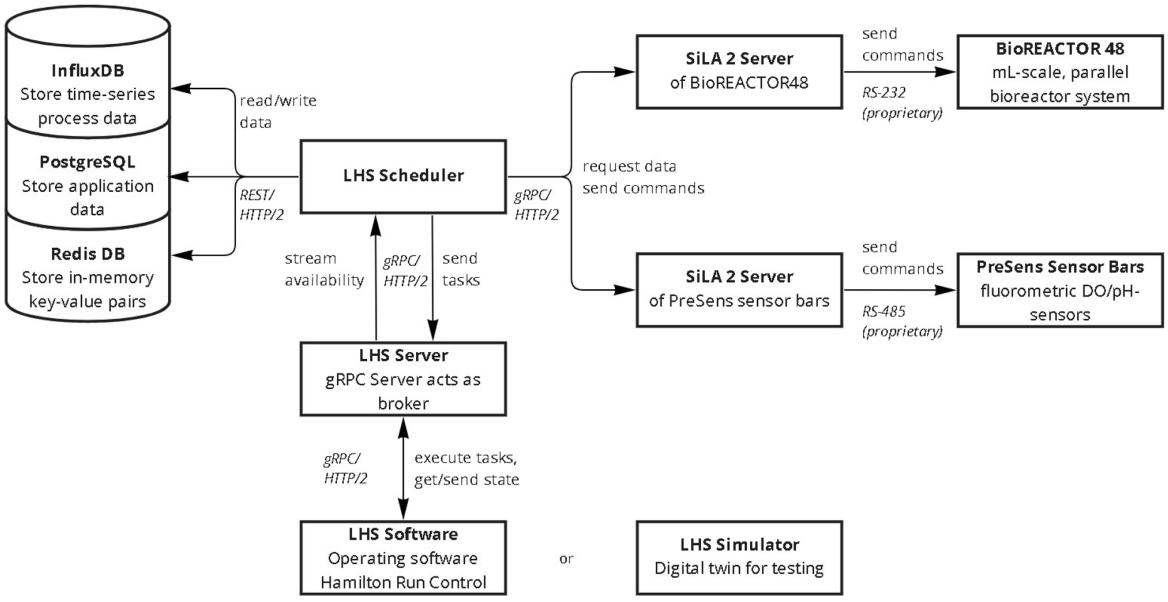

Fig. 2 The proposed software solution consists of several elements: The central scheduler application, the SiLA 2 device servers for the parallel bioreactor system and the pH/DO sensor bars, the gRPC server for LHS communication. Furthermore, it relies on three databases for the storage of experiment process data (InfluxDB), the persisting of application data (PostgreSQL) and for in-memory storage and inter-thread and process communication (RedisDB)

system, the sensor bars, the LHS, as well as the evaluated and processed data such as task priorities, task execution times, and setpoints. The stored data are enriched with timestamps and meta data such as experiment name, reactor position, and operator to enable good data management and data filtering for, e.g., real-data visualization or data export for subsequent analyses. The LHS Scheduler uses the python package *influxdb* to communicate with the database server. If no dedicated InfluxDB server is provided to the application, a fall-back database will be created using the official InfluxDB image hosted on DockerHub (*influxdb* v.1.7.11). Process data are visualized in real time using the

complimentary web-service application Chronograf (Chronograf v.1.8.5, InfluxData, San Francisco).

PostgreSQL

PostgreSQL (PostgreSQL v.6.0.9, PostgreSQL Global Development Group) is a widely adopted SQL database. The python package *psycopg2* (v.2.8.6) is used to communicate with the database from within the LHS Scheduler application. If no dedicated PostgreSQL server is available, a fall-back database is created. This database is run in a docker container using the official PostgreSQL docker image available on dockerhub (*postgres* v.13). The complimentary database management system pgAdmin4 (pgAdmin4 v.5, pgAdmin Development Team) is used for data access and visualization of the application data.

LHS server

Pipetting stations are common island solutions and integration into other third-party software can be difficult as software interfaces are oftentimes missing, hidden, or unexposed. In most cases, external control is not required or not intended. For the proposed scheduling software however, enabling external control is a critical requirement. HSL, deriving from C, supports the execution of code elements that were registered on the operating system as Component Object Model (COM). COM interop is a technology developed by Microsoft to enable interoperability between COM-libraries and the Windows.NET Framework in the Common Language Runtime (CLR) [36, 37]. This allows the registration of C# code to be registered as COM-Interop and subsequently to be imported into the VENUS method editor library. With this procedure, an interface was created based on a C# gRPC client that was introduced into the VENUS environment as shown in Fig. 3.

The information transmitted between the LHS Scheduler and the LHS runtime environment via the LHS Server is kept as simple as possible including only the essential information required for execution. This encompasses the name of the method, an ordered array containing the volumes to be transferred, as well as the respective target and, if required, source sequence. Target and source sequences are transferred as lists (Python) and arrays (C#) and are mapped to the labware definition within Venus. This ensures that only existing positions are addressed.

LHS simulator

To improve the development process and increase the software quality, a digital twin of the LHS has been developed to mimic the behavior for *in silico* software testing: The LHS Simulator. The LHS simulator receives commands from the

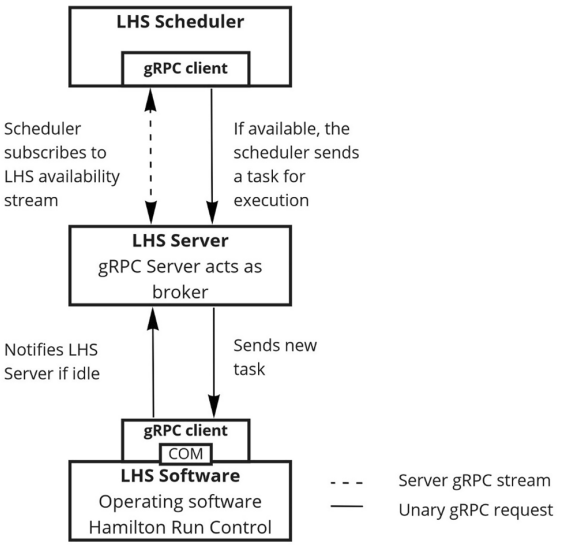

Fig. 3 External control of the LHS is realized by introducing a C# gRPC client into the VENUS environment by registering it as COM-Interop and including it into the VENUS method library. The dedicated LHS gRPC Server acts as broker between the scheduling software and the VENUS runtime. The scheduler incorporates a python gRPC client to communicate with the gRPC server, hence controlling the LHS by sending execution request and receiving the responses transferred by the LHS Server. At the same time, the LHS Server is aware of the LHS's state and shares this information with the scheduler software through a gRPC stream. This leads to an event-based execution with minimal delay and downtime

LHS Server, interprets and checks their validity regarding constraints and execution order, and simulates the execution time based on historical data. The LHS simulator loads the user script at the start of a simulation run and evaluates the specifications supplied by the user. During runtime, the scheduler output is compared against the user input. Discrepancies between the specifications in the user script and the output regarding parameters like violated constraints, missed executions, or miscalculated feeding volumes are stored are detected and stored in a log file. The LHS Simulator is based on a gRPC Client and uses the same interface description as the C# client of the LHS. The source code is written in python.

Dynamic scheduling software

The scheduler core

The LHS Scheduler is based on a gRPC server. Once started, the application is configured by an experiment-specific user script. Within this user script, the information about the experiment, the used time-series database, the involved

devices, and the tasks and their respective configuration is provided. This information is persisted in the PostgreSQL database as application data via invocation of the corresponding function of the application programming interface (API). The API is a useful tool for testing and development as it allows the interaction with the scheduler application during runtime. However, this is not required nor intended for the regular use case presented in Control of Parallelized Bioreactors II.

A scheduler instance consists of three phases: initialization, run, and termination. During initialization, the information on the experiment, the specified tasks, the required devices and databases is loaded. At this stage a device initialization sequence is run that sets up a device connection and configures them correctly for use by the application. At the end of the initialization procedure, threads for data acquisition and the initially scheduled tasks are started.

During a fermentation process, the scheduler application continuously communicates with the task threads and keeps track of their reported priorities and status. Furthermore, the data acquisition threads are monitored and restarted if needed. Within the run-loop, tasks can be added or removed from the scheduler, which will spawn or kill the respective task threads. Furthermore, the run-loop subscribes to a stream provided by the LHS Server, to keep track of the status of the LHS. The availability of the LHS is communicated via this stream in real time. If available, the scheduler will dispatch the task with the highest priority to the LHS Server for execution. The scheduler instance and the task threads all rely on the internal clock of the scheduler instance main loop. For simulation purposes, the internal time can be manipulated for fast run simulations. In this mode, the LHS Simulator is used and the device drivers of the bioreactor system and the sensor bars are set to simulation mode.

The termination sequence is a safety mechanism that is executed in the following two scenarios: (1) The scheduler shuts down in a controlled way either by invoking the respective API command or by the execution of a scheduled abort task or (2) if the scheduler main loop exits unexpectedly. The termination sequence stops all running task threads, closes the stream channel with the LHS Server, stops the device data acquisition, and executes a device termination sequence. This is particularly important, if devices are used that may take or cause damage in case of an unexpected loss of control. The device termination sequence will break out of the run-loop and resume in the LHS Scheduler main loop.

Data acquisition

The dynamic scheduling capability is dependent on the availability of real-time data. Before entering the scheduler run-loop, the LHS Scheduler starts a data acquisition thread for each data source. The data sources are defined in the user script and, thus, the API. A data source is defined by their SiLA 2 connection details, the measurement intervals and, similar to the scheduler run-loop, respective data acquisition scripts for initialization, run, and termination. The data is stored in the InfluxDB and accessible to the other threads for processing and priority calculation.

The process data can be accessed either directly by using an InfluxDB database client or the web-based visualization tool Chronograf as shown in Fig. 4. The LHS Scheduler application and its task and data acquisition threads write their data into the InfluxDB database. Each data point consists of a measurement name, tags like experiment name and reactor positions, a timestamp, and fields containing the values of the parameters that make up that measurement. A data point is unique. Specific data points or subsets thereof can be visualized or exported using the SQL-like query language InfluxQL as shown in Fig. 4.

Apart from defining the database, the retention policy and the measurement of interest, the query can be filtered by tags, timestamps and a combination thereof. The queries in Fig. 4 fetch and visualize the dissolved oxygen (top) and pH (bottom) data of 48 reactor positions over the duration of 24 min of an *E. coli* fed-batch process described in Control of Parallelized Bioreactors II. The operations executed by the LHS at that time are displayed in Fig. 5.

Task threads

To run fermentation experiments in mL-scale bioreactors that are integrated into LHS systems, several tasks must be performed by the LHS. These tasks include, but are not limited to, substrate addition, pH control, induction, and a sampling task that can consist of multiple sub-steps (for example: sampling, sample preparation, plate reader measurement, cleaning of MTP). The LHS Scheduler executes the task with the highest priority. Priorities are calculated by the tasks threads themselves based on repeated evaluation of the available real-time data, such as the current pH in every reactor at the current time. A task for DO-control utilizing the available real-time DO data and the access to the bioreactor stirrer was not implemented as DO-control was not required by the application use case presented in the second part (Control of parallelized bioreactors II).

Task threads and their specific implementation may vary between use cases. To allow for the creation of new tasks or enable adaptation of existing tasks, the task thread class inherits from a base class that implements all functions that are shared between tasks.

Running tasks threads communicate with the LHS Scheduler run-loop via queue objects to report their current priority. If the scheduler selects the task for execution, the task thread is informed about the start- and end time of the event. Task thread data such as the thread runtime, the time of the

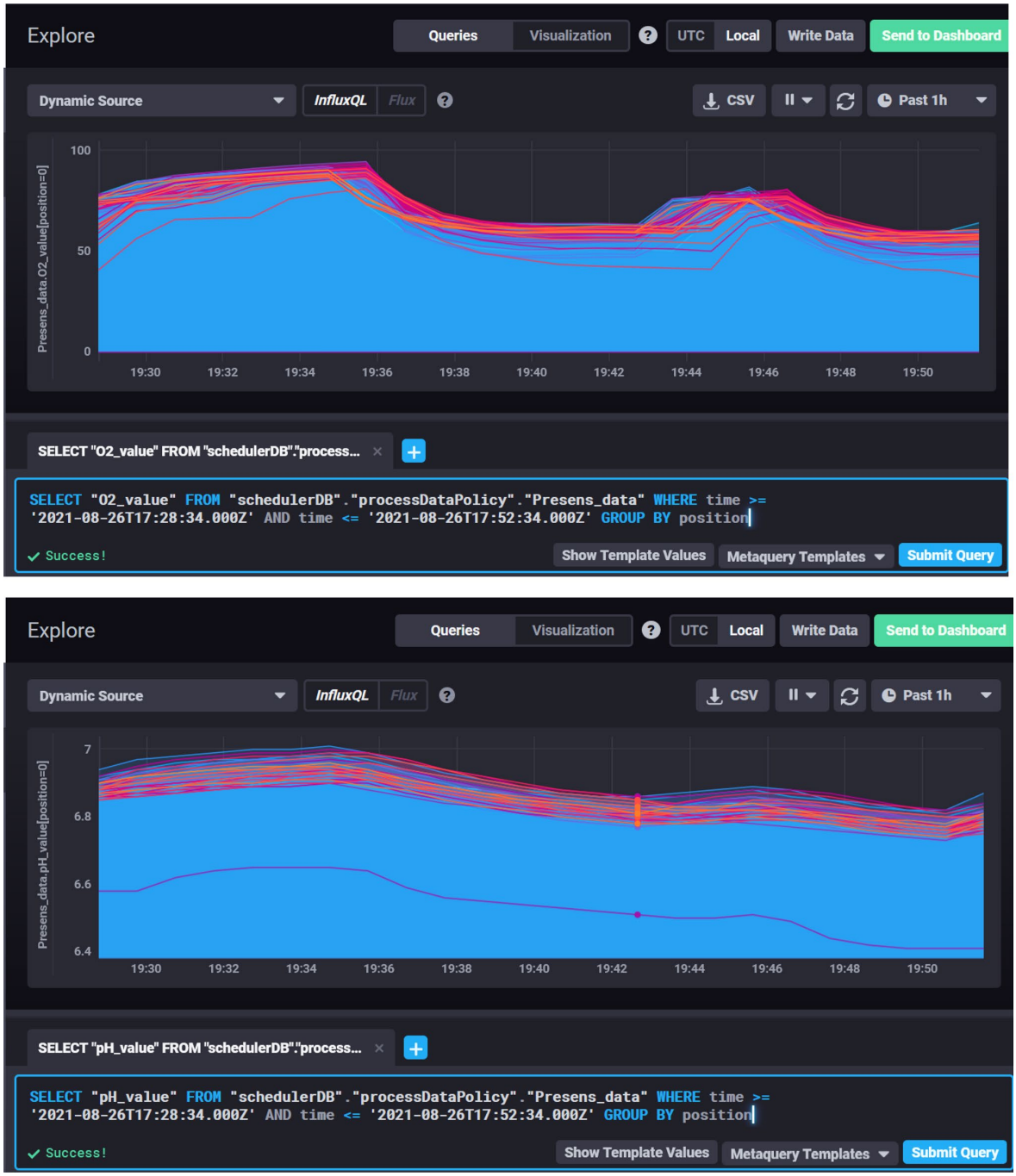

Fig. 4 The browser-based graphical user interface Chronograf connects to the InfluxDB and allows for real-time data visualization. In this figure, the process parameters dissolved oxygen (top) and pH (bottom) are shown for the same excerpt of an example process in parallel stirred-tank bioreactors

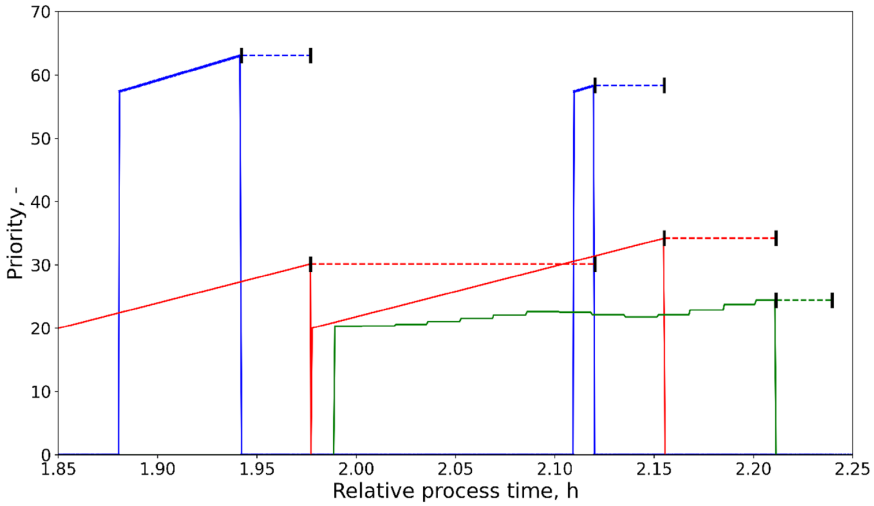

Fig. 5 The priority changes of varying tasks are shown: the feeding task (−, blue line), stages 4 and 5 of the sampling task (−, red line), and the pH-control task (−, green line). The execution times of these tasks are marked by dashed lines in the respective color with bars representing the start and end time. The relative process time is identical to the example excerpts of the process data shown in Fig. 5. The substrate solution contained 300 g L^{-1} glucose. Cultivation conditions and process parameters are described in detail in the second part (Control of Parallelized Bioreactors II)

last execution, the priority as well as set points and present values are written frequently to the influx database with the respective experiment and task-specific tags.

Priority calculation

The execution of a task depends on external factors such as physical limitations and the design of the experiment like fixed execution times for an induction operation. These constraints must be evaluated before any priority calculation. Constraints are very task specific and require the in-depth knowledge of the process and the involved machines. Pipetting robots may, for example, not be able to pipette extremely small or large volumes due to physical limitations of the pipetting channels. Furthermore, a process with a substrate addition task must be performed on an interval that is itself constrained by upper and lower time boundaries so that the deviation from that interval remains minimal. Hence, there are time and volume constraints. While some of these constraints are soft and may be violated, such as an upper time constraint of a substrate addition task, some are hard constraints that prohibit execution, such as the minimum volume that can be pipetted.

Constraints are reactor specific, vary from task to task and must be specified by the user. A python class is provided that fully defines a constraint. A task thread will use the constraint evaluation function to assess each constraint and whether and to what extent, each constraint is violated and output a vector containing a boolean for each reactor. The task thread will resume with the priority calculations only if no hard constraints are violated.

All task thread objects use the same priority calculator class in which the priority calculation algorithms are defined. A task thread contains an instance of the priority manager and passes all relevant data to this object, such as the algorithm to be used and the number of active reactor positions. The number of tasks that require the shared resource, the dispensing unit, is low. However, the issue lies within the bottleneck situations in which multiple executions are necessary at the same time. This requires the optimization of the execution order based on priorities that are calculated based on real-time data. It was found that this can be achieved with simple static and linear functions as priority algorithms.

The priority calculator differentiates between no priority, a base priority and a critical priority. A task that either violates a hard constraint or does not require execution has a priority equal to zero. If the need for execution is evaluated, the priority calculation has access to the parameters p_{base} and p_{crit}. The priority of a single task cannot exceed its critical priority:

$$p_{\text{max}} = p_{\text{crit}}.$$

The implemented priority calculation functions are:

Step

A static priority determination that depends on constraint evaluation alone. If a task is within its constraint limits, it is executable. As soon as it starts exceeding its soft boundaries, the priority will be raised.

$$p = \begin{cases} 0, & \text{if hard constraint violated} \\ p_{\text{base}}, & \text{if no constraint violated} \\ p_{\text{crit}}, & \text{if soft constraint violated.} \end{cases}$$

Step specific

The priorities are broken down to the relative contributions of the affected reactor positions that require action (n_{aff}) respective to the number of total active reactor positions (n_{all}).The number of total active reactors may vary as reactor positions with malfunctions are automatically excluded by the LHS Scheduler.

$$p\left(n_{\mathrm{aff}}, n_{\mathrm{all}}\right) = \begin{cases} 0, & \text{if hard constraint violated} \\ p_{\mathrm{base}} \dfrac{n_{\mathrm{aff}}}{n_{\mathrm{all}}}, & \text{if no constraint violated} \\ p_{\mathrm{base}} + \left(p_{\mathrm{crit}} - p_{\mathrm{base}}\right) \dfrac{n_{\mathrm{aff}}}{n_{\mathrm{all}}}, & \text{if soft constraint violated.} \end{cases}$$

Dynamic volume

The priority is a linear function of the volume and the number of active positions and the contributions towards the priority are weighted per reactor positions. This function is aimed at tasks in which the execution depends on volume, such as the required volume of substrate in a feeding task or the volume of base to be added in a pH-control task. The vector v contains the feed or base volumes of each reactor position that fulfills the constraints. The vector $v_{\mathrm{lim,lower}}$ contains the lower volume limit that can be added for each position and Δv_{lim} contains the allowed volume space between the lower and upper volume limit of each position ($\Delta v_{\mathrm{lim}} = v_{\mathrm{lim,upper}} - v_{\mathrm{lim,lower}}$).

$$p\left(n_{\mathrm{all}}, n_{\mathrm{aff}}, v\right) = \begin{cases} 0, \text{ if hard constraint violated} \\ p_{\mathrm{base}} + \dfrac{p_{\mathrm{crit}} - p_{\mathrm{base}}}{n_{\mathrm{all}}} \displaystyle\sum_{i=1}^{n_{\mathrm{aff}}} \dfrac{v_i - v_{i,\mathrm{lim},lower}}{\Delta v_{i,\mathrm{lim}}}, \\ \text{if no constraint violated} \end{cases}.$$

Dynamic time

The priority is a linear function of the time and the number of active positions. The contributions of each reactor position are weighted. This function is aimed at tasks in which the execution is time dependent, such as sampling or induction. The vector t contains the time since the last execution of each reactor position that fulfills the constraints. The remaining vector notation is transferable from the previous method.

$$p\left(n_{\mathrm{all}}, n_{\mathrm{aff}}, t\right) = \begin{cases} 0, & \text{if hard constraint violated} \\ p_{\mathrm{base}} + \dfrac{p_{\mathrm{crit}} - p_{\mathrm{base}}}{n_{\mathrm{all}}} \displaystyle\sum_{i=1}^{n_{\mathrm{aff}}} \dfrac{t_i - t_{i,\mathrm{lim,lower}}}{\Delta t_{i,\mathrm{lim}}}, \\ quad & \text{if no constraint violated.} \end{cases}$$

Figure 5 displays the behavior in a bottleneck situation in which multiple tasks require access to the LHS. In this example, a simple linear dynamic priority calculation dependent on time (dynamic time: sampling) or volume (dynamic volume: feeding and pH control) was applied. As a result, the feeding task was executed with the minimal possible feeding interval. Furthermore, the last sampling step could be executed swiftly as pH regulation was not required urgently, with a, at times, even decreasing priority (also see pH visualization in Fig. 4). The decreasing priority of pH can be explained by a switch of metabolism in the *E. coli* cultures from overflow metabolism to glycogen metabolism [38]. The first substrate addition (Fig. 5) leads to overflow metabolism and acetate production, lowering the pH continuously (Fig. 4). Surpassing the threshold of the pH control, the lower volume constraint of base, the priority rises. The switch to glycogen metabolism and the corresponding consumption of acetate increases the pH and, thus, lowers the priority of the pH-control task.

Conclusion and outlook

The proposed software enables the dynamic scheduling of LHS operations required for bioprocess control in parallel bioreactor systems, leading to an improved task execution order compared to conventional proprietary vendor software. The user can create tasks with pre-defined execution intervals allowing experiments in which these parameters are of utmost importance, like studies on scalability regarding protein expression or population heterogeneity, while relying on simple, transparent and editable priority calculation algorithms.

The acquisition of all process data by the LHS Scheduler software in a central database reduces the risk of information loss or error and enables real-time visualization of the data. Storing data of multiple sources in a central database makes the data readily available during the process and afterwards, simplifying data processing and analysis. The data generated in regular optimization processes is time-series data. However, if data sources such as chromatography data or data stemming from image analyses are to be acquired, time-series databases would not be well suited, as they are not

optimized for such data types. In such cases, the setup of a data lake should be considered in future work.

The increasing degree of complexity of automated fermentation setups results in an increasing number of points of failure. As this is not desirable, testing of software and experimental plans is important. While software malfunctions can be prevented by unit, integration, and end-to-end tests, the actual simulation of experimental runs remains complicated when dealing with biological systems. As proposed, the execution of *in silico* experiments prior to the experimental run can be achieved by simulating experimental conditions with digital twins of the devices involved. Thus, future work should aim towards improving code quality by extending the existing testing infrastructure but also by improving the digital twin implementations of the bioreactor system, the sensor bars and the LHS to achieve more realistic *in silico* experiment simulations by, e.g., the integration of mechanistic models or the use of empirical data.

The LHS Server opens a backdoor to take external control of a Hamilton STARlet LHS and thus the integration of such systems into greater automation workflows. Other approaches aim towards full external control, thus transferring LHS responsibilities like tracking of lab ware positions to the user [39, 40]. This approach is error prone and complex because it requires intrinsic knowledge of the LHS as it circumvents the method editor supplied by the vendor. The proposed solution is a good compromise between the two, as it leverages the benefits of external control as well as the proprietary method editor.

The presented software consists of the LHS Scheduler, Server, and Simulator as well as the SiLA 2 device servers. They are written in Python, making them easily readable and adjustable to changing requirements. The general structure of the LHS Scheduler enables the expansion with new tasks or priority calculation algorithms. Future work should include the development of a graphical user interface to increase usability and enable human intervention in the running scheduling process, like adjusting task definitions during runtime.

The following software components are provided on request by the authors: LHS Scheduler, LHS Server, LHS Simulator. The SiLA 2 device drivers for the BioREACTOR48 and the Presens sensor bars are publicly available in the following GitLab repository: https://gitlab.com/biovt/sila2lib_implementations

Acknowledgements The authors thank the German Ministry of Education and Research (BMBF) for funding within the national joint research project Digitalization in Industrial Biotechnology (DigInBio). Grant number: FKZ 031B0463B. Nikolas von den Eichen and Lukas Bromig thank for the support provided by the TUM Graduate School (Technical University of Munich, Germany).

Author contributions Lukas Bromig, Nikolas von den Eichen, and Dirk Weuster-Botz contributed to the conception and design of the study; Supervision and funding acquisition by Dirk Weuster-Botz; Lukas Bromig and Nikolas von den Eichen designed, realized and validated the software; Lukas Bromig provided the figures and the original draft of the manuscript. All the authors contributed to the critical revision and final approval of the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was funded by the German Ministry of Education and Research (BMBF) within the national joint research project Digitalization in Industrial Biotechnology (DigInBio, grant number 031B0463B). Open Access funding enabled and organized by Project DEAL.

Code availability The SiLA 2 device drivers are publicly available in the following GitLab repository: https://gitlab.com/biovt/sila2lib_implementations

Declarations

Conflict of interest Dirk Weuster-Botz is Editor-in-Chief of Bioprocess and Biosystems Engineering. One of the Associate Editors of the journal was assigned to assume responsibility for overseeing peer review.

References

1. Kusterer A, Krause C, Kaufmann K, Arnold M, Weuster-Botz D (2008) Fully automated single-use stirred-tank bioreactors for parallel microbial cultivations. Bioprocess Biosyst Eng 31(3):207–215. https://doi.org/10.1007/s00449-007-0195-z
2. Schmideder A, Hensler S, Lang M, Stratmann A, Giesecke U, Weuster-Botz D (2016) High-cell-density cultivation and recombinant protein production with Komagataella pastoris in stirred-tank bioreactors from milliliter to cubic meter scale. Process Biochem 51(2):177–184. https://doi.org/10.1016/j.procbio.2015.11.024
3. Lattermann C, Büchs J (2015) Microscale and miniscale fermentation and screening. Curr Opin Biotechnol 35:1–6. https://doi.org/10.1016/j.copbio.2014.12.005
4. Von den Eichen N, Bromig L, Sidarava V, Marienberg H, Weuster-Botz D (2021) Automated multi-scale cascade of parallel stirred-tank bioreactors for fast protein expression studies. J Biotechnol 332:103–113. https://doi.org/10.1016/j.jbiotec.2021.03.021
5. H. Chmiel and D. Weuster-Botz, 'Bioreaktoren', in *Bioprozesstechnik*, Springer, 2018, 157–229.
6. Marques MP, Szita N (2017) Bioprocess microfluidics: applying microfluidic devices for bioprocessing. Curr Opin Chem Eng 18:61–68. https://doi.org/10.1016/j.coche.2017.09.004
7. Weuster-Botz D, Puskeiler R, Kusterer A, Kaufmann K, John GT, Arnold M (2005) Methods and milliliter scale devices for

high-throughput bioprocess design. Bioprocess Biosyst Eng 28(2):109–119. https://doi.org/10.1007/s00449-005-0011-6
8. Anane E, Sawatzki A, Neubauer P, Cruz-Bournazou MN (2019) Modelling concentration gradients in fed-batch cultivations of *E. coli* - towards the flexible design of scale-down experiments: Modelling concentration gradients in fed-batch. J Chem Technol Biotechnol 94(2):516–526. https://doi.org/10.1002/jctb.5798
9. Puskeiler R, Kusterer A, John GT, Weuster-Botz D (2005) Miniature bioreactors for automated high-throughput bioprocess design (HTBD): reproducibility of parallel fed-batch cultivations with Escherichia coli. Biotechnol Appl Biochem 42(3):227–235. https://doi.org/10.1042/BA20040197
10. Tajsoleiman T, Mears L, Krühne U, Gernaey KV, Cornelissen S (2019) An industrial perspective on scale-down challenges using miniaturized bioreactors. Trends Biotechnol 37(7):697–706. https://doi.org/10.1016/j.tibtech.2019.01.002
11. Bareither R, Pollard D (2011) A review of advanced small-scale parallel bioreactor technology for accelerated process development: Current state and future need. Biotechnol Prog 27(1):2–14. https://doi.org/10.1002/btpr.522
12. Junne S, Neubauer P (2018) How scalable and suitable are single-use bioreactors? Curr Opin Biotechnol 53:240–247. https://doi.org/10.1016/j.copbio.2018.04.003
13. Achinas S, Heins J-I, Krooneman J, Euverink GJW (2020) Miniaturization and 3D printing of bioreactors: a technological mini review. Micromachines. https://doi.org/10.3390/mi11090853
14. Ladner T, Grünberger A, Probst C, Kohlheyer D, Büchs J, Delvigne F (2017) 15—application of mini- and micro-bioreactors for microbial bioprocesses. Curr Develop Biotechnol Bioeng. https://doi.org/10.1016/B978-0-444-63663-8.00015-X
15. Gebhardt G, Hortsch R, Kaufmann K, Arnold M, Weuster-Botz D (2011) A new microfluidic concept for parallel operated milliliter-scale stirred tank bioreactors. Biotechnol Prog 27(3):684–690. https://doi.org/10.1002/btpr.570
16. Morschett H et al (2021) Robotic integration enables autonomous operation of laboratory scale stirred tank bioreactors with model-driven process analysis. Biotechnol Bioeng 118(7):2759–2769. https://doi.org/10.1002/bit.27795
17. Faust G, Janzen NH, Bendig C, Römer L, Kaufmann K, Weuster-Botz D (2014) Feeding strategies enhance high cell density cultivation and protein expression in milliliter scale bioreactors. Biotechnol J 9(10):1293–1303. https://doi.org/10.1002/biot.201400346
18. Haby B et al (2019) Integrated robotic mini bioreactor platform for automated, parallel microbial cultivation with online data handling and process control. SLAS Technol Transl Life Sci Innov 24(6):569–582. https://doi.org/10.1177/2472630319860775
19. Nickel DB, Cruz-Bournazou MN, Wilms T, Neubauer P, Knepper A (2017) Online bioprocess data generation, analysis, and optimization for parallel fed-batch fermentations in milliliter scale. Eng Life Sci 17(11):1195–1201. https://doi.org/10.1002/elsc.201600035
20. Schmideder A, Cremer JH, Weuster-Botz D (2016) Parallel steady state studies on a milliliter scale accelerate fed-batch bioprocess design for recombinant protein production with Escherichia coli. Biotechnol Prog 32(6):1426–1435
21. Meo A, Priebe XL, Weuster-Botz D (2017) Lipid production with Trichosporon oleaginosus in a membrane bioreactor using microalgae hydrolysate. J Biotechnol 241:1–10
22. Teworte S, Malcı K, Walls LE, Halim M, Rios-Solis L (2022) Recent advances in fed-batch microscale bioreactor design. Biotechnol Adv 55:107888. https://doi.org/10.1016/j.biotechadv.2021.107888
23. Lemoine A, Delvigne F, Bockisch A, Neubauer P, Junne S (2017) Tools for the determination of population heterogeneity caused by inhomogeneous cultivation conditions. J Biotechnol 251:84–93
24. Fernandes RL et al (2011) Experimental methods and modeling techniques for description of cell population heterogeneity. Biotechnol Adv 29(6):575–599
25. Heins A-L, Weuster-Botz D (2018) Population heterogeneity in microbial bioprocesses: origin, analysis, mechanisms, and future perspectives. Bioprocess Biosyst Eng 41(7):889–916. https://doi.org/10.1007/s00449-018-1922-3
26. de Jonge LP et al (2011) Scale-down of penicillin production in Penicillium chrysogenum. Biotechnol J 6(8):944–958
27. Lemoine A, Maya Martínez-Iturralde N, Spann R, Neubauer P, Junne S (2015) Response of Corynebacterium glutamicum exposed to oscillating cultivation conditions in a two- and a novel three-compartment scale-down bioreactor. Biotechnol Bioeng 112(6):1220–1231. https://doi.org/10.1002/bit.25543
28. Puskeiler R, Kaufmann K, Weuster-Botz D (2005) Development, parallelization, and automation of a gas-inducing milliliter-scale bioreactor for high-throughput bioprocess design (HTBD). Biotechnol Bioeng 89(5):512–523. https://doi.org/10.1002/bit.20352
29. SiLA 2, 'SiLA 2 Part (A)—overview, concepts and core specification'. Dec 21, 2020, [Online]. https://docs.google.com/document/d/1nGGEwbx45ZpKeKYH18VnNysREbr1EXH6FqlCo03yASM/edit (accessed Jan 18, 2021).
30. SiLA 2, 'SiLA 2 Part (B)—mapping specification', *Google Docs*. https://docs.google.com/document/d/1-shgqdYW4sgYIb5vWZ8xTwCUO_bqE13oBEX8rYY_SJA/edit?usp=embed_facebook (accessed Jan 24, 2022).
31. SiLA, 'SiLA2/sila_base', *GitLab*, 2018. https://gitlab.com/SiLA2/sila_base (accessed Dec 21, 2020).
32. 'SiLA2 / sila_python', *GitLab*. https://gitlab.com/SiLA2/sila_python (accessed Jan 24, 2022).
33. L. Bromig, F. Moorhof, and N. von den Eichen, 'SiLA 2 Service Implementations', *GitLab*. https://gitlab.com/biovt/sila2lib_implementations/-/tree/master (accessed Mar. 03, 2022).
34. J. Shahid, *InfluxDB Documentation*. Release, 2019.
35. S. N. Zehra, 'Time Series Databases and InfluxDB', p. 45.
36. Barnaby T (2002) Understanding COM Interop. In: Barnaby T (ed) Distributed .NET Programming in VB .NET. Apress, Berkeley, CA, pp 273–288
37. Microsoft Corporation, .'NET Framework documentation on MSDN - COM Interop'. https://docs.microsoft.com/en-us/dotnet/visual-basic/programming-guide/com-interop/ (accessed Jan. 25, 2022).
38. Nyström T (1994) The glucose-starvation stimulon of Escherichia coli: induced and repressed synthesis of enzymes of central metabolic pathways and role of acetyl phosphate in gene expression and starvation survival. Mol Microbiol 12(5):833–843. https://doi.org/10.1111/j.1365-2958.1994.tb01069.x
39. Chory EJ, Gretton DW, DeBenedictis EA, Esvelt KM (2021) Enabling high-throughput biology with flexible open-source automation. Mol Syst Biol 17(3):e9942. https://doi.org/10.15252/msb.20209942
40. E. J. Chory, D. W. Gretton, E. A. De Benedictis, K. M. Esvelt, 'Flexible open-source automation for robotic bioengineering', 2020. doi: https://doi.org/10.1101/2020.04.14.041368.

6 Accelerated Adaptive Laboratory Evolution by Automated Repeated Batch Processes in Parallelized Bioreactors

This publication showcases another example of how a modular automation and digitalization infrastructure enables flexible automation that accelerates R&D in industrial biotechnology. Adaptive Laboratory Evolution (ALE) has emerged as a powerful tool to advancing strain engineering and microbial research. It allows organisms to adapt to specific environmental conditions, enabling the discovery of beneficial mutations that improve traits such as stress tolerance, tolerance to toxic substances, substrate utilization, production, and growth efficiency. Industrial applications of ALE span diverse fields, including biofuels, pharmaceuticals, and food production [Mavrommati *et al.* 2022; Sandberg *et al.* 2019]. Recent publications are focusing increasingly on automated methods that provide better data in terms of quality and quantity [Chen *et al.* 2024; Halle *et al.* 2023; Orsi *et al.* 2024; Zuchowski *et al.* 2023]. Such approaches not only deepen our understanding of microbial evolution, but also provide valuable insights for engineering strains tailored for sustainable bio-based economies.

This publication presents a novel method for ALE that utilizes automated repeated batch processes in parallelized bioreactors. This approach transitions from traditional serial passaging in shake flasks to a more efficient L-scale stirred-tank bioreactor system, significantly reducing manual labor and overcoming technical limitations associ-

ated with shake flask experiments. The method allows for real-time monitoring of key process variables as well as the application of soft sensor technology for increased experimental insights such as biomass generation, accelerating ALE experiments. The study demonstrates that this method is 9.4 times faster than comparable shake flask experiments in the reference literature and 3.6 times faster than automated mL-scale methods, as evidenced by growth optimization experiments with *E. coli* K-12 MG1655 using glycerol as a carbon source.

The research highlights the influence of scale, process control, and passage size on evolutionary progress, and challenges reference literature regarding the cumulative number of cell divisions (CCD) as sole measure for ALE speed with it not only being dependent on the CCD, but also on the scale and control of the experiment, as well as the number and size of passages. In addition, the paper discusses the application of ALE in industrial biotechnology, emphasizing the benefits of automation and parallelization to improve the efficiency of evolutionary processes.

The implementation presented in this publication builds on the modular automation infrastructure established in previous works. The SiLA 2 Manager, developed as a microservice-based integration layer [Bromig *et al.* 2022a], served as the core software platform for realizing the ALE experiments. Its modular architecture enabled the reuse of existing components, including the SiLA 2-compliant connector applications for the parallel bioreactor system[1], off-gas analysis[2], and peristaltic pumps[3] [Bromig *et al.* 2022a,b; Von den Eichen *et al.* 2021]. These examples demonstrate the benefit of adopting SiLA 2 as a standard for device communication and underline the reusability and extensibility of the integration framework. The ability to deploy complex, real-time process control strategies for ALE with minimal additional software development

[1]DASGIP 4X Parallel Bioreactor System, DASGIP AG, part of Eppendorf AG, Hamburg Germany

[2]BlueVary, BlueSens gas sensor GmbH, Herten, Germany

[3]Masterflex Ismatec Reglo ICC, Avantor, Inc., Pennsylvania, United States

highlights the long-term scalability of the system architecture.

In this publication, I was responsible for the conceptualization, design, and implementation of the automated ALE approach. I developed the complete software infrastructure that enabled parallelized repeated batch processes in L-scale stirred-tank bioreactors, including the integration of real-time process monitoring, control, and soft sensor functionality for biomass estimation. I designed and conducted all experiments, curated and analyzed the data, and generated all visualizations. I also wrote the original draft of the manuscript and coordinated the project administration in collaboration with the supervising co-author.

microorganisms

MDPI

Article

Accelerated Adaptive Laboratory Evolution by Automated Repeated Batch Processes in Parallelized Bioreactors

Lukas Bromig and Dirk Weuster-Botz *

Chair of Biochemical Engineering, Technical University of Munich, Boltzmannstraße 15, D-85748 Garching, Germany
* Correspondence: dirk.weuster-botz@tum.de

Abstract: Adaptive laboratory evolution (ALE) is a valuable complementary tool for modern strain development. Insights from ALE experiments enable the improvement of microbial cell factories regarding the growth rate and substrate utilization, among others. Most ALE experiments are conducted by serial passaging, a method that involves large amounts of repetitive manual labor and comes with inherent experimental design flaws. The acquisition of meaningful and reliable process data is a burdensome task and is often undervalued and neglected, but also unfeasible in shake flask experiments due to technical limitations. Some of these limitations are alleviated by emerging automated ALE methods on the µL and mL scale. A novel approach to conducting ALE experiments is described that is faster and more efficient than previously used methods. The conventional shake flask approach was translated to a parallelized, L scale stirred-tank bioreactor system that runs controlled, automated, repeated batch processes. The method was validated with a growth optimization experiment of *E. coli* K-12 MG1655 grown with glycerol minimal media as a benchmark. Off-gas analysis enables the continuous estimation of the biomass concentration and growth rate using a black-box model based on first principles (soft sensor). The proposed method led to the same stable growth rates of *E. coli* with the non-native carbon source glycerol 9.4 times faster than the traditional manual approach with serial passaging in uncontrolled shake flasks and 3.6 times faster than an automated approach on the mL scale. Furthermore, it is shown that the cumulative number of cell divisions (CCD) alone is not a suitable timescale for measuring and comparing evolutionary progress.

Keywords: adaptive laboratory evolution (ALE); *Escherichia coli*; process development; repeated batch process; process automation; growth rate optimization; glycerol utilization, biomass estimator, black box model, batch process

Citation: Bromig, L.; Weuster-Botz, D. Accelerated Adaptive Laboratory Evolution by Automated Repeated Batch Processes in Parallelized Bioreactors. *Microorganisms* **2023**, *11*, 275. https://doi.org/10.3390/microorganisms11020275

Academic Editor: Stefan Junne

Received: 8 November 2022
Revised: 12 January 2023
Accepted: 13 January 2023
Published: 20 January 2023

1. Introduction

Adaptive laboratory evolution (ALE) is increasingly gaining relevance in industrial biotechnology as a tool for improving production strains [1]. Insights gained by adaptive laboratory evolution through whole-genome sequencing can be directly applied to the fields of metabolic engineering (reverse engineering) and complemented with classical genetic engineering to yield improved microbial production systems [2–6]. Most publications focus on the evolutionary aspects or applications of ALE, which have been reviewed extensively [1,7–9]. The techniques and experimental setups are increasing in their degree of automation and sophistication as shown by Radek et al. (2017), who translated ALE to an automated microtiter plate format, or Wang et al. (2020), who used a microbial microdroplet culture platform [10–13]. Wong et al. (2018) and Ekkers et al. (2020) described a low-cost automated batch system on the mL scale and LaCroix et al. (2017) used an automated batch system on the mL scale for ALE growth experiments of *E. coli* on glycerol [14–16]. However, automation efforts have so far been focused on miniaturization and parallelization but not on optimization, process control, and data quality. ALE methods for chemostat cultures

have a different field of application and were not considered in this study [10,12]. The serial passaging of batch cultures, a tedious and labor-intensive task, is the most popular ALE method [8]. However, the way serial passaging is performed brings several disadvantages that impact the outcome of ALE experiments and are highly inefficient.

Designing the right stress for the sought after target is a crucial part of ALE experiment design. Studies that aim to improve the microbial growth rate or growth with non-native carbon sources rely on the serial passaging of batch experiments. A major driver for improving growth rates of microbial cell factories with alternate carbon sources is the increasing market pressure to utilize surpluses or cheaper substrates in bioprocesses that produce fine and bulk chemicals that compete with classical chemical processes [17] and the general aim to improve space–time yields. In these cases, the specific growth rate is used as the metric of fitness since the efficiency of carbon source utilization is rate-limiting and directly coupled to growth [8,18]. Propagation during the exponential phase and a reduction in or even elimination of a lag phase become important as a prolonged stationary phase or lag phase may shift the evolutionary pull towards an undesired fitness archetype, thus allowing other traits to be selected for [8,16,19,20].

Furthermore, genetic evolution is sensitive to the process conditions; thus, state variables must be controlled tightly [21]. Process control, the norm in liter scale stirred-tank bioreactors, is limited in shake flask cultures. The classical manual approach of serial passaging in shake flasks results in a suboptimal stress design that has become accepted in the scientific community. Comparable automated ALE batch experiments operating on the mL scale have limited capabilities for process control, i.e., a lack of pH and DO control, as well as the continuous data acquisition of key state variables [14,16]. The dynamic nature of batch cultivations results in specific growth rates, cell densities, and environmental conditions that change continuously not only during a single batch but, inherently in ALE processes specifically, also between each sequential batch. This extra level of dynamics has one major drawback: the process time of the batches decreases over time and the process time can only be estimated from the results of the previous batch. In order to maintain a consistent schedule, individual batches are artificially prolonged to fit a predefined, usually daily, passaging schedule by adjusting the initial cell concentration to fit a forecasted 24 h process. However, this results in at least one of the following problems: (1) the initial cell concentration can affect the duration of the lag phase [22–24] and a very small passage size may lead to beneficial mutations being lost [16]. Hence, the initial cell concentration should be large enough and kept constant between batches, which also results in lag-phase minimization, which is beneficial for ALE experiments targeting the growth rate and growth on non-native carbon sources. (2) The cells enter the stationary phase for an unknown duration before the passaging, making the calculation of the specific growth rate for this batch unreliable and inducing the wrong selection stress as the organism is cultivated in growth-limiting conditions. Lee et al. suggested the cumulative number of cell divisions (CCD) as a meaningful parameter for measuring the evolutionary speed in ALE experiments and that fundamental kinetics can be derived [25]. However, deriving transferable insights from such a multivariate experiment is complex and using CCD as such a metric has multiple limitations. The CCD does not take into account the number of performed passages, the passage size, or other factors that depend on the reactor system, such as the reactor volume.

In this study, a comparative analysis was performed in which experiments from Lee et al. (2011) were reproduced in the proposed automated ALE experimental setup. The results were compared with experimental data from manual and automated systems [2,16,25,26]. It is further shown that an alternative experimental setup can greatly improve the efficiency at which these experiments are conducted, not just regarding the timescale of evolution but also by reducing the impact of undesired stresses caused by prolonged stationary and lag phases or large temporal changes in initial biomass concentrations.

2. Material and Methods

2.1. Bacterial Strain

The experiments were carried out using fresh cultures of the wild-type strain *E. coli* K12 MG1655 from the German Collection of Microorganisms and Cell Cultures (#DSM 18039, DSMZ GmbH, Braunschweig, Germany). The same wild-type strain was used in the reference literature [2,25,26]. The freeze-dried culture was prepared in LB medium according to DSMZ instructions and stored as cryo-stock at −80 °C in 50% glycerol.

2.2. Media

Seed cultures were grown with lysogeny broth (LB) medium containing, per liter, 10 g peptone, 5 g yeast extract, and 5 g NaCl, which was adjusted to pH 7 with NaOH. The repeated batch processes were conducted with medium according to Riesenberg et al. (1991) (RB) containing, per liter, 8.4 mg EDTA, 8.4 mg $CoCl_2 \cdot 6\,H_2O$, 15 mg $MnCl_2 \cdot 4\,H_2O$, 1.5 mg $CuCl_2 \cdot 2\,H_2O$, 3 mg H_3BO_3, 2.5 mg $Na_2MoO_4 \cdot 2\,H_2O$, 13 mg $Zn(CH_3COO)_2 \cdot 2\,H_2O$, 0.1 g Fe(III) citrate, 13.3 g KH_2PO_4, 4 g $(NH_4)_2HPO_4$, 1.7 g citric acid $\cdot\,H_2O$, 2.4 g NaOH, 1.2 g $MgSO_4 \cdot 7\,H_2O$, and the carbon source glycerol at a concentration of 15 g L^{-1} [27]. Anti-Foam (AF 204, Sigma-Aldrich, Merck KGaA, Darmstadt, Germany) was added at a concentration of 1 mL L^{-1}. The RB medium used in experiments with mutagen contained an additional 5 mg L^{-1} of 1-methyl-3-nitro-1-nitrosoguanidine (NTG, CAS-#: 70-25-7, Biosynth s.r.o, Bratislava, Slovak Republic). All media were sterilized in an autoclave at 120 °C for 20 min. NTG was sterile-filtered and added to the media flasks just prior to process start. A list with additional information on used chemicals and media preparation is provided in the Supplementary Materials.

2.3. Seed Culture

A total of 0.1 mL inoculum (cryo-stock) was added to 100 mL of autoclaved LB medium in baffled 500 mL Erlenmeyer flasks. Seed cultures were incubated in a rotary shaker (Multitron, Infors AG, Bottmingen, Switzerland) at 37 °C and 150 rpm up to 12 h until a final cell concentration of OD_{600} of 2–2.5 was reached. If necessary, cell suspensions were concentrated by an additional centrifugation and resuspension step to obtain the required inoculum concentration.

2.4. Experimental Setup and Process Control

Batch cultivations were performed in 4 parallel stirred-tank bioreactors (DASGIP Parallel Bioreactor System, Eppendorf AG, Hamburg, Germany) on an L scale, each with a total working volume of 575 mL. Autoclaved RB medium, including the carbon source glycerol, was used for the repeated batch processes to enable unlimited growth conditions. The proposed process achieves higher cell densities and, to avoid limitations, RB medium with a higher nitrogen, sulfur and phosphorus content, compared to the M9 medium used in related works [2,16,25,26], was used. The composition of the RB medium used in this study is closely related to the M9 medium used in the reference cultivations. Kangwa et al. (2015) showed that both synthetic, mineral media (RB, M9) result in the same biomass production and, thus, that the impact on growth rate is expected to be negligible [28]. The initial inoculation was performed manually to yield an initial biomass concentration of 0.2 g L^{-1} in the individual stirred-tank bioreactors, whereas the inoculation of the sequential batches was performed indirectly as described in the following section about the reactor setup, resulting in measured initial biomass concentrations between 0.15–0.4 g L^{-1}. The pH was controlled using 12.5% (v/v) NH_4OH and 2 M HCl, which were added via the peristaltic pumps of the reactor system to maintain pH 7. The bioreactor was aerated with an air flow set between 1.16–3.48 vvm. Processes were started with a stirrer rate of 600 rpm and regulated up to speeds of 1200 rpm. Dissolved oxygen (DO) levels and pH were measured continuously using a DO probe (Visiferm DO ECS 225 H0, Hamilton Bonaduz AG, Bonaduz, Switzerland) and a pH probe (EasyFerm Plus PHI K8 225, Hamilton Bonaduz AG, Bonaduz, Switzerland). The pH probe was calibrated before autoclaving by

2-point calibration and the DO probe was calibrated before process start by first aerating the medium with pure nitrogen (DO = 0%) followed by aeration with pressured process air until equilibrated (DO = 100%). The reactors were equipped with a PT-100 temperature probe (Platinum RTD temperature sensor, Eppendorf AG, Hamburg, Germany) and a level sensor (Level Sensor, Eppendorf AG, Hamburg, Germany). Off-gas was condensed and analyzed for on-line measurement of CO_2 and O_2 (BlueVary, BlueSens gas sensor GmbH, Herten, Germany).

ALE experiments were performed by passaging in a repeated batch mode for up to 200 h. The experimental setup is shown in Figure 1. Experiments were conducted with glycerol as a carbon source with and without the mutagen NTG to create experiments comparable to the ones presented by Fong et al. (2005), Herring et al. (2006), Lee et al. (2011), and LaCroix et al. (2017) [2,16,25,26] and to validate the acceleration potential of the proposed setup to a manual and another automated method. Cultivations were performed at 37 °C, the same temperature at which the experiments of Lee et al. (2011) were executed at. In contrast, the set of experiments conducted by Fong et al. (2005), which were used by Herring et al.(2006), were performed at 30 °C. LaCroix et al. (2017) did not specify a cultivation temperature in their publication or Supplementary Materials. These limitations must be considered when directly comparing the presented results with the data derived from the experiments of Fong et al. (2005) and LaCroix et al. (2017). ALE experiments were stopped manually once a stable phenotype, i.e., the specific growth rate, was detected in at least three consecutive batches by photometric at-line measurements. Samples of the last evolution population were taken, diluted with 50% glycerol, and stored in a freezer at −80 °C.

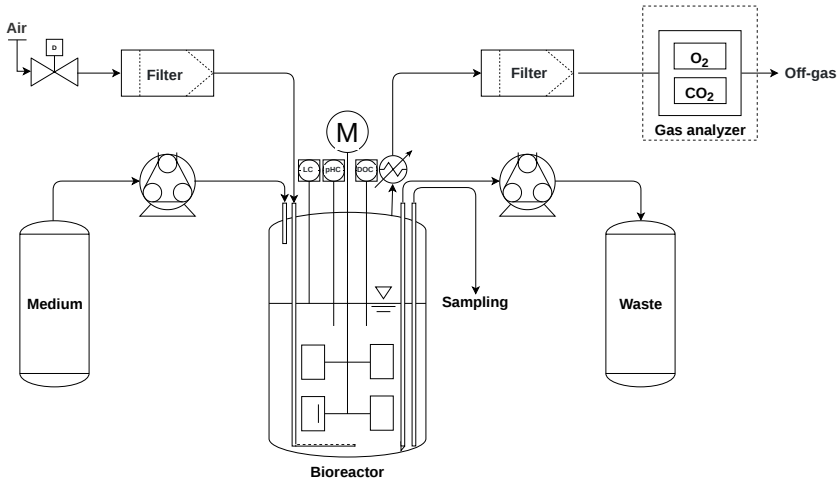

Figure 1. A simplified scheme of the experimental setup with one bioreactor is shown. Experiments were performed in four parallelized stirred-tank bioreactors. Sterile, pressured air was sparged at the bottom of the bioreactor and, upon exit, condensed at the bioreactor gas outlet by a reflux condenser, subsequently sterile-filtered, and passed through the off-gas analyzer. The RB medium was provisioned in 5 L flasks, which were exchanged with fresh medium every eighth batch. Removed biosuspension was pumped into a 50 L shared waste container. The bioreactor was filled and emptied by peristaltic pumps. A rising pipe with a beveled needle point style end (bevel angle of 90°) was constructed to empty the bioreactor adequately. In-line probes for medium exchange-control, pH-control, and DO-control were inserted through the bioreactor head. Two Rushton turbines (Rushton-Type stirrer, Eppendorf, Hamburg, Germany) were placed on the stirrer axis starting 5 mm above the end of the axis, with a vertical distance of 25 mm between the turbine wings.

2.4.1. Off-Gas Analysis

The concentrations of CO_2 and O_2 were analyzed using an external off-gas analyzer (BlueVary, BlueSens gas sensor GmbH, Herten, Germany), which was integrated into the

external process control software using a driver that was developed as part of this work according to the Standardization in Laboratory Automation 2 device interface standard (SiLA 2).

2.4.2. Peristaltic Pumps

Two peristaltic pumps with four individually controllable channels (Ismatec Reglo ICC, Cole- Parmer, Wertheim, Germany) were used to pump out the biosuspension and refill the bioreactors with fresh medium. Thus, each bioreactor required two individually controllable pump channels. The pumps were integrated and controlled by the external process control software via a SiLA 2 driver developed for this pump as part of this work.

2.4.3. Process Control Software

The proprietary process control software (DASware® Control, Eppendorf AG, Hamburg, Germany) was used for the basic batch process control, such as pH control, DO control, and temperature control. Process control logic related to ALE and passaging was performed using the workflow and scripting environment of an external control software (SiLA 2 Manager) that was developed in previous works [29]. To enable the integration of the reactor system into an external control software, the proprietary vendor software DASWare® Connect is required, which provides an OPC-UA interface. As part of this work, a driver was developed that translates this interface into a standardized interface following the Standardization in Laboratory Automation specification (SiLA 2, Rapperswil-Jona, Switzerland). The integration of all devices in the external control software via the standardized interfaces enabled the acquisition of all relevant process data in a central database (InfluxDB, InfluxData Inc., San Francisco, CA, USA). Process data of the bioreactor system and the off-gas analyzer were acquired in 15 *s* intervals. A control script, written in the external control software's Python editor (Supplementary Materials), orchestrated the passaging process between the batch phases by accessing and controlling the bioreactor system as well as the peristaltic pumps. All drivers were written in Python using the SiLA 2 Python reference implementation [30].

2.4.4. Repeated Batch Mode

Figure 2 shows exemplary process data to explain the applied control scheme of the passaging process. Passaging is triggered automatically at the end of the exponential growth phase (I), which is detected by a sudden spike in DO due to depletion of the carbon source (glycerol). The pump action is triggered if a DO signal threshold of 75% air saturation is surpassed. In order to avoid faulty executions during the early stages of the batch process in which a DO >75% is to be expected, a minimal batch time of 1 h was defined, during which, this mechanism is deactivated. The peristaltic pumps in Figure 1 are connected to the bioreactor head via reliable, sterile quick-couplings (PMCD220212 and PMCD170212, Colder Products Company, Mörfleden-Walldorf, Germany). Five-liter glass flasks (DU218017304, DWK Life Sciences GmbH, Wertheim Germany) were used for media storage, which had to be replaced every eighth batch. A rising pipe with a sharp phased-off end was confectionized for the waste stream so that the bioreactor could be emptied almost entirely. Temperature and pH control were switched off during the transfer process (II + III, Figure 2) to avoid erroneous controller behavior after dry running of the sensors. Agitation and aeration were continued during the draining process (II). During the refill process (III), agitation was stopped, whereas the aeration was continued to ensure a sterile environment and minimal oxygen availability. The bioreactor level sensors were calibrated to a working volume of 575 mL before the start of the ALE experiment and provided the input for the stop trigger of the peristaltic pumps during the refill process. As shown in Figure 2, phase III, the pumping action was stopped once a conductivity of greater than 200 mS was detected by the level sensor, which translates to the earliest contact between medium and probe. Each bioreactor was controlled separately.

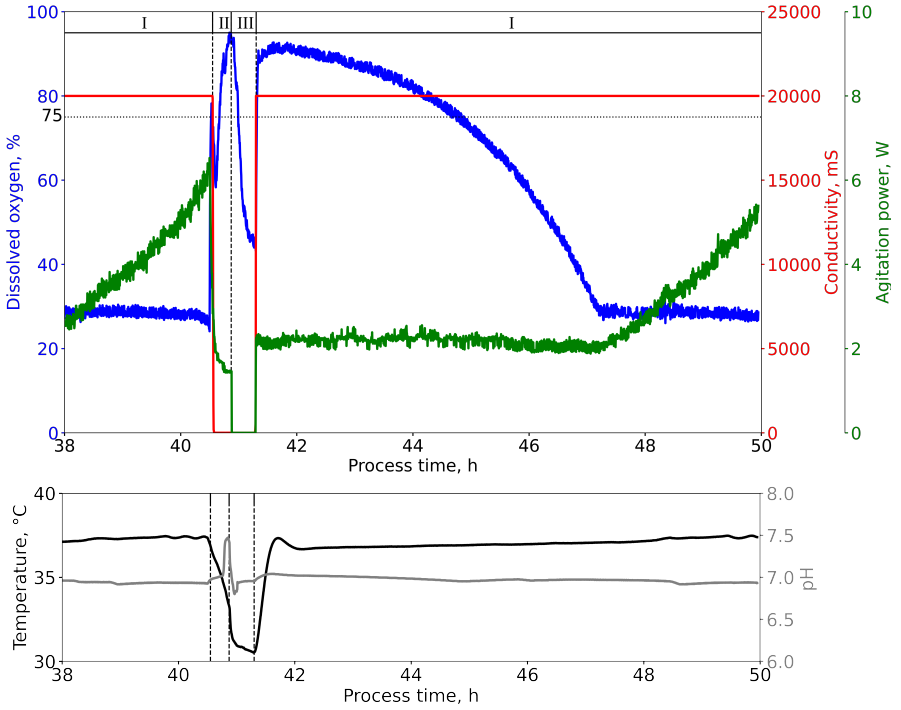

Figure 2. Process data of the transition between two consecutive batches is shown and classified into three distinct phases: the batch phase (I), the draining phase (II), and the refill process phase (III). The transition from phase (I) to (II) is triggered by a DO signal that surpasses a set threshold of 75 % air saturation. The duration of the draining process is dependent on the maximum calibrated pump rate of the respective channel. The pump is addressed with the pump rate by volume mode and parametrized with the bioreactor volume, including an additional buffer volume to ensure complete removal of the biosuspension. The transition from phase (II) to (III) is triggered by a rise in the conductivity signal over a fixed threshold of 200 mS. Agitation is automatically down-regulated during phase (II) and switched off during phase (III).

2.5. Specific Growth Rate Estimation Using a Black-Box Model

2.5.1. Black Box Model

To describe the change in biomass and substrate concentration over the course of the ALE experiment and to derive process parameters such as specific growth rate and CCD, the captured on-line data were used as input for a black-box model. The conversion rates (r_i in mol of i/h) are derived from a simplified metabolic black-box equation for aerobic *E. coli* growth:

$$r_s \cdot CH_{2.67}O + r_o \cdot O_2 + r_n \cdot NH_3 \longrightarrow r_x \cdot CH_{1.77}O_{0.49}N_{0.24} + r_c \cdot CO_2 + r_w \cdot H_2O \quad (1)$$

An elemental biomass composition $CH_{1.77}O_{0.49}N_{0.24}$ of an *E. coli* K12 strain was used [31,32]. The model was simplified with the assumptions of aerobic growth on a single carbon and energy source, and without product formation. In addition to the mass balances, a degree of reduction (DoR) balance was used. The degree of reduction in the used elements can be defined as follows: $\gamma_N = -3$, $\gamma_C = 4$, $\gamma_{O_2} = -4$, $\gamma_H = 1$ [32]. As this definition leads to a DoR of 0 for NH_3, r_N does not appear in either of the applied balances and can remain undetermined. Hence, the equation system remains over-determined, as there are two balances (C and O) for the calculation of one conversion rate (r_X). Normal-

izing all carbon-containing species on a 1 C-mole basis, the following equation system with the respective yield coefficients ($Y_{i,j}$ in mol of i/mol of j) was obtained:

$$C\text{-}balance : 1 + Y_{x,CO_2} - Y_{x,gly} = 0 \tag{2}$$

$$H\text{-}balance : 1.77 + Y_{x,H_2O} \cdot 2 - Y_{x,\mathrm{gly}} \cdot 2.67 - Y_{x,NH_3} \cdot 3 = 0 \tag{3}$$

$$O\text{-}balance : 0.49 + Y_{x,CO_2} \cdot 2 + Y_{x,H_2O} - Y_{x,\mathrm{gly}} - Y_{x,O_2} \cdot 2 = 0 \tag{4}$$

$$N\text{-}balance : 0.2 - Y_{x,NH_3} = 0 \tag{5}$$

$$DoR\text{-}balance : 4.19 + Y_{x,\mathrm{gly}} \cdot 4.67 - Y_{x,O_2} - 4 = 0 \tag{6}$$

2.5.2. Oxygen Uptake Rate and Carbon Emission Rate

The oxygen uptake rate (OUR) and carbon emission rate (CER) were calculated from the air flow into the bioreactor $F_{a,in}$, the bioreactor volume V_r, the molar volume of an ideal gas at 25 °C V_m, and the gas fractions $y_{i,in/out}$ of O_2 and CO_2 in the in- and outflow stream according to Equations (7) and (8). The volumetric fractions of CO_2 and O_2 of the inflow gas are assumed constant at $y_{O_2,in} = 0.041$ and $y_{CO_2,in} = 20.91$, respectively. Accumulation of oxygen or carbon dioxide in the fermentation broth can be neglected as their solubility is a function of temperature and pH, which is controlled and assumed constant. Therefore, the OUR and CER are equal to the respective conversion rate r_o and r_c.

$$r_o = OUR = \frac{F_{a,in}}{V_m \cdot V_r} \cdot \left(-y_{O_2,in} + \frac{1 - y_{O_2,in} - y_{CO_2,in}}{1 - y_{O_2,out} - y_{CO_2,out}} \right) \cdot y_{O_2,out} \tag{7}$$

$$r_c = CER = \frac{F_{a,in}}{V_m \cdot V_r} \cdot \left(-y_{CO_2,in} + \frac{1 - y_{O_2,in} - y_{CO_2,in}}{1 - y_{O_2,out} - y_{CO_2,out}} \right) \cdot y_{CO_2,out} \tag{8}$$

2.5.3. Biomass and Substrate Conversion Rate

Equations (9) and (10) are derived from the equation system presented in the Equations (1)–(6). Coupled with Equations (7) and (8), the conversion rates for biomass (Equation (9)) and substrate (Equation (10)) can be estimated over the course of the batch process.

$$r_x = -8.32 \cdot r_o - 9.71 \cdot r_c \tag{9}$$

$$r_s = -8.32 \cdot r_o - 8.71 \cdot r_c \tag{10}$$

The C-mole normalized molecular mass of the substrate is $M_{gly} = 30.701$ g mol^{-1} and that of the biomass is $M_{gly} = 24.996$ g mol^{-1}. They are used to convert the conversion rates from a C-mole basis to gram. Since the cell specific growth rate $\mu = r_x$, the following equation can be formulated for the biomass concentration c_x.

$$c_x = \frac{1}{\mu} \cdot \frac{dc_x}{dt} \tag{11}$$

Equation (11) is integrated numerically by trapezoidal rule using the *trapz()* function of the *NumPy* Python library with an initial concentration value of $c_{x,0} = 2.5$ g L^{-1}. The specific growth rate μ was calculated within the exponential growth phase. To avoid the impact of the lag phase, only the last 60% of the batch was taken into account when calculating the specific growth rate. A more detailed description of the model is given in the Jupyter notebook provided in the Supplementary Materials.

2.5.4. Cumulative Number of Cell Divisions

Cell concentrations are calculated using the average molecular weight of *E. coli* $m_{E.\ coli}$ with the aforementioned cell composition. The number of cells can be calculated with following relationship:

$$N = \frac{c_x \cdot V_r}{m_{E.coli}} \tag{12}$$

where c_x is the biomass concentration in g L^{-1}, V_R is the reactor volume in L, and $m_{E.coli}$ is the dry mass of a single *E. coli* cell in g. The average cell mass of $m_{E.coli} = 2.90 \times 10^{-13}$ g from the literature was used [33]. The same cell mass was also used for calculation of CCD in the reference literature [25]. The number of generations was calculated per batch i according to Equation (13) and used for the calculation of the cumulative number of cell divisions CCD using Equation (14), where n_i is the number of generations in batch i, $N_{0,i}$ is the number of cells at the start, and N_i the number of cells at the point of interest, i.e., the end of the batch. The cumulative number of cell divisions is defined as the sum of the cell divisions of m consecutive batches [25].

$$n_i = \frac{log(N_i/N_{0,i})}{log2} \tag{13}$$

$$CCD = \sum_{i=1}^{m} N_{0,i}(2^{n_i} - 1) \tag{14}$$

2.5.5. Preprocessing

Process data captured by the parallel bioreactor system and the off-gas analyzer were stored in the InfluxDB and subsequently downloaded as .csv file. The time-series datasets of each bioreactor were separated into the respective batch datasets using a pattern detection algorithm that identifies the process phases as introduced in Figure 2. To smooth out short-term signal fluctuations, a moving average was applied with a window width of n = 6 measurements of 15 s using the *rolling()* function of the Python software library *pandas*. The datasets of the CO_2 and O_2 concentrations of each batch were fitted to an exponential function of the type $f(x) = a \cdot e^{b \cdot x} + c$ using the *optimize.curve_fit()* function of the Python *SciPy* library. Datasets with noisy or irregular process data resulting in a fit with an R^2 below 0.98 were discarded and not evaluated. A baseline correction was performed to set the y intercept of the fit to the concentration of the inflow of the respective gas for all batches after the first batch to make up for sensor shift over the course of the experiment and for unreliable data within the first 10 min after the reactor content replacement.

3. Results

3.1. Process Characteristics and Duration

The proposed method was validated by executing ALE growth rate optimization experiments of *E. coli* K12 MG1655 on the non-native carbon source glycerol with the mutagens NTG and a control group without the mutagens NTG and comparing the findings with the results of the reference literature [2,25,26]. Experiments with glycerol and mutagen were named GM1, GM2, and GM3, and experiments with glycerol only were named G1, G2, and G3. The automated ALE experiments were conducted for up to 200 h, resulting in a total of 19–25 sequential batches, after which, they were terminated manually. Between batches, an average reactor volume exchange duration of 42.22 ± 3.86 min was recorded, which was dependent on the maximum possible speed of the respective channels of the peristaltic pumps, which varied between 25–28 mL min^{-1}.

Figure 3 shows CO_2 and O_2 off-gas data of one of the experiments, with NTG highlighting the apparent continuous decrease in the process duration over the course of the experiment. ALE batch duration of *E. coli* cultures without NTG decreased by 40.7% from 11.17 h at batch 2 to 6.62 h at batch 22, whereas, in cultures with NTG, the average batch

duration decreased by 45% from 10.29 ± 0.71 h at batch 2 to 5.67 ± 1.31 h at batch 22. The process data are provided in the Supplementary Materials.

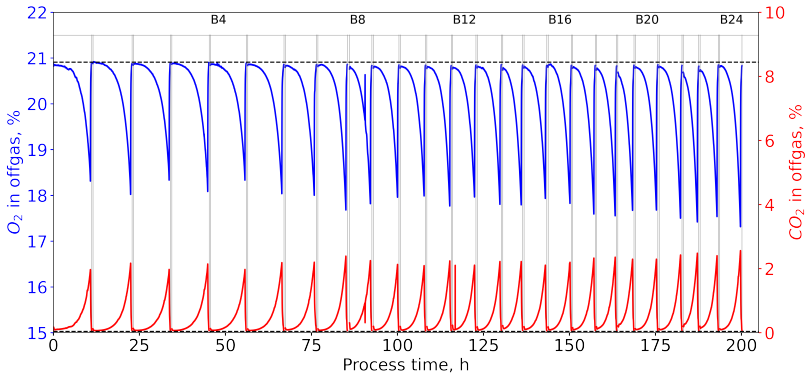

Figure 3. The concentrations of O_2 and CO_2 in the off-gas of an automated ALE experiment (GM2) in an L scale stirred-tank bioreactor are shown over the course of 25 consecutive batch experiments totaling a process duration of 200 h. A culture of *E. coli* K12 MG1655 was grown with RB medium at 37 °C, 600–1400 rpm, 40 vvm, and an initial glycerol concentration of 12 g L^{-1} as sole carbon source. The ALE process was performed in an automated system in a repeated batch mode with a bioreactor volume of 575 mL. Vertical lines indicate the start and end of the medium exchange procedure between batches (grey). The concentrations of $O_2 = 20.91\%$ and $CO_2 = 0.04\%$ in the pressurized air in the inflow are depicted by the horizontal, dashed lines (black). Batch numbers are indicated with B4–B24. The shown data were used as input for the black box model to calculate OUR and CER, as well as derived state variables, such as the estimated biomass and substrate concentrations and the specific growth rate according to Equation (7)–(10).

3.2. Model Predictions of Specific Growth Rate and Biomass Concentrations

Figure 4 shows the logarithm of the estimated biomass concentrations of all batches of the ALE experiment GM2 depicted in Figure 3 as calculated from the black box model introduced in Equations (1)– (6). Early batches experience a long lag and transition phases before entering fully exponential growth. This effect is greatly reduced as cultures evolve towards improved glycerol metabolization and growth. Calculating specific growth rates over the full batch duration would be impacted severely by this effect. Hence, the specific growth rates presented in the following sections were calculated based on the last 50% of the respective batches data, only taking into account the linear slope of the curves.

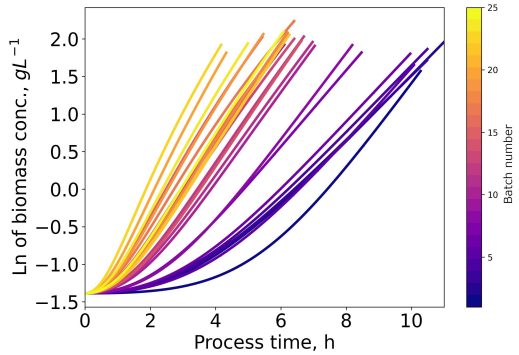

Figure 4. The logarithm of the biomass concentration is depicted over the process time. Each line represents a single batch of the same ALE experiment (GM2). The chronological order of the batches is indicated by the color bar. Early batches show an extended period with a non-linear relationship (lag), which decreases in length as the culture evolves towards a stable phenotype.

3.3. Comparison of Serial Passaging and Repeated Batch Process

Specific growth rate results from the ALE experiments that were estimated using the black-box model were validated with the results from at-line DCW measurements (see Supplementary Materials). DCW measurements of selected batches were taken in 2 h intervals. Figure 5 shows the relative fitness of ALE experiments with and without mutagens plotted against the CCD. The calculated specific growth rates were put into the relation of the wild-type stable phenotype to derive a relative fitness parameter. The average specific growth rate of the strains that evolved in the presence of NTG was 0.70 ± 0.05 h^{-1} compared to 0.61 ± 0.03 h^{-1} of the culture without the mutagens, which was used as a reference for the calculation of the relative fitness parameter. The key parameters of Figure 5 are summarized in Table 1 and compared with the reference studies. It was observed that the WT cultures with the native mutation rate (G1, G2, G3) arrive at a stable phenotype faster but exhibit a lower stable specific growth rate compared to cultures that were grown with the additional mutagens NTG (GM1, GM2, GM3). While NTG cultures reached comparable specific growth rates as the control group in a similar timeframe at a $log_{10}(CCD) = 14.1$, the adaption process continued further and reached a stable specific growth rate that was 14.8% higher than that of the control with the native mutation rate.

The results of the stable specific growth rates of both experimental groups are within the margin of error of the published reference experiments [2,25,26]. However, the CCD required to attain the stable phenotype differs greatly between the experiments of this study and the literature reference. Whereas the manual serial passaging method results in a total of 3.9×10^{11} CCDs, the repeated batch method requires a total CCD of 1.26×10^{14} in cultures of the WT *E. coli* without NTG. A similar order of magnitude is observed in cultures with NTG. In the reference literature in which a manual system was used, the stable specific growth rates were obtained after a total of 1056 h in cultures without NTG [26]. LaCroix et al. (2017) investigated the effect of different passage sizes (0.001–10%) on growth rate adaptation in their automated system. The presented results (passage size 2–4%) are compared with their results of the group with passage size 1%, which plateaued at a comparable growth rate of 0.60 ± 0.01 h^{-1} after 456 h.

In the presented work, the overall process duration of the repeated batch experiments to reach the similar specific growth rate result was achieved in 127.3 ± 13.6 h without NTG, and 160.8 ± 5.6 h with NTG cultures, respectively. This equates to a reduction in the process duration by 85–88% if compared to the 44-day-long culturing by serial passaging without NTG by Fong et al. (2005) and a 69–75% reduction compared to the 19 days required with the culturing process in the automated system by LaCroix et al. (2017). However, a comparison with Fong et al. (2005) must be interpreted with the caveat that a culture temperature of 30 °C was used instead of 37 °C as in the presented use case.

Table 1. Comparison of the key results of the ALE experiments performed in this study compared to the results from the literature.

Evolution	Method	Replicates	GR (h^{-1})	CCD	Ref.
WT	mL scale, manual	5 (GA, GB, GC, GD, GE)	0.64 ± 0.04	$3.9 \times 10^{11} \pm 0.5 \times 10^{11}$	[2,26]
	mL scale, automated	6 (1% passage size)	0.60 ± 0.01	$1.57 \times 10^{12} \pm 0.14 \times 10^{12}$	[16], suppl.
	L scale, automated	3 (G1, G2, G3)	0.61 ± 0.03	$1.26 \times 10^{14} \pm 0.25 \times 10^{14}$	this study
WT + NTG	mL scale, manual	2 (GM1, GM2)	0.74	$0.93 \times 10^{11} \pm 0.01 \times 10^{11}$	[25]
	L scale, automated	3 (GM1, GM2, GM3)	0.70 ± 0.05	$2.45 \times 10^{14} \pm 0.21 \times 10^{14}$	this study

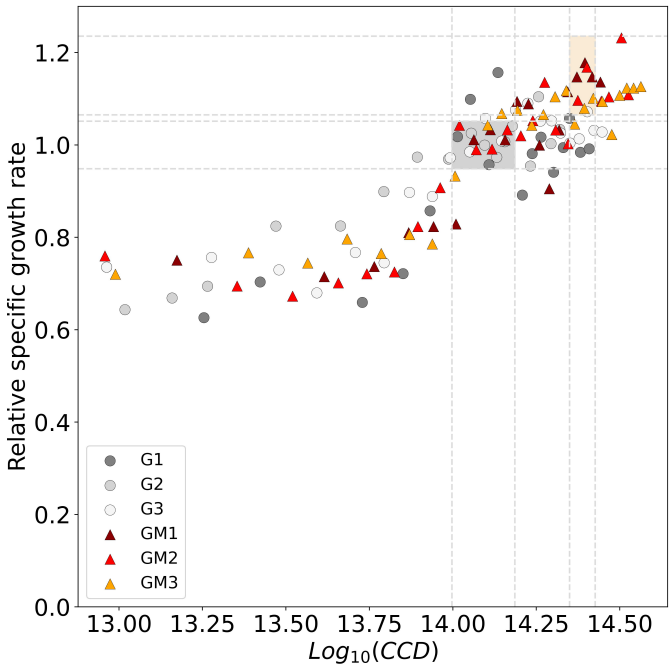

Figure 5. The relative fitness of replicate ALE experiments with *E. coli* K12 MG1655 growing with the non-native carbon source glycerol. Relative fitness is defined as the stable specific growth rate divided by the average stable specific growth rate of the control group of WT *E. coli* without NTG. The cumulative number of cell divisions (CCD) is used as time scale to measure adaptation progress. The specific growth rate is considered stable if the moving average of three consecutive batches has an absolute standard deviation < 0.01 h^{-1} and no further upwards trend. This definition of a stable phenotype is specific to this set of experiments. A decisive criterion was required to make near-real-time decisions while the experiment was running; hence, a variability based approach was chosen that focuses on the change in optimization metric: the specific growth rate. The specific growth rate of the stable phenotype of the *E. coli* wild-type cultures without NTG is 0.61 ± 0.03 h^{-1} at a $log_{10}(CCD) = 14.09 \pm 0.09$ and is used to compare both groups and calculate the relative fitness (grey shaded area, relative fitness of the WT $= 1 \pm 0.05$). The observed average specific growth rate of the WT strain with NTG is 0.70 ± 0.05 h^{-1} and was reached at a $log_{10}(CCD) = 14.39 \pm 0.04$ (orange shaded area, relative fitness of the WT with NTG $= 1.15 \pm 0.08$).

4. Discussion

Adaptive laboratory evolution experiments are laborious and time-consuming. A few automated ALE systems, operating on the microliter and milliliter scale, have been proposed in the past [10,12,14,16]. These systems are typically based on a microtiter plate format that enables biomass determination by measuring OD_{600} with a plate reader. Systems with reaction vessels on the mL scale are agitated with magnetic stir bars and provide OD measurements either with a respective sensor or via at-line measurements with a liquid handling station and plate reader. ALE systems that operate on the microliter and milliliter scale can be parallelized at a lower cost, but are limited to low biomass densities and restricted with regard to process control [14–16]. The lack of pH and DO control or the acquisition of the respective data leads to changing and unknown process conditions that directly impact the stresses exerted on the culture. The proposed system is agitated, aerated, and enables appropriate process control, resulting in transparent and reliable culture conditions. However, the quality of the data acquired could be further improved in the future by using OD probes for biomass determination to obtain an additional source for real-time biomass concentration data.

The successful modeling of batch processes is highly dependent on the quality of the state parameters and the starting conditions as the error is propagated and aggregated over time. Hence, the assumption of a constant biomass concentration at each batch start introduces a small error into the system. Furthermore, it was observed that CO_2 concentrations in the off-gas at the batch start spiked briefly before going down to expected levels. The effect was observed only in subsequent batches but not the first, and is most likely caused by the stripping of CO_2 from the fresh medium, whereas the medium of the first batch was already equilibrated during the DO-calibration procedure before the process start. An exponential fit was performed on the data to reduce the impact of this spike. However, going forward, the over-determination of the equation system could be used and further state variables, such as dissolved oxygen or pH, could be added to extend the model with a data reconciliation strategy to reduce the error of the estimate and even quantify the prediction confidence. A similar strategy has been applied successfully in soft sensors for biomass estimation in fed-batch processes [34,35].

The continuous acquisition of process data allows for the calculation of the estimated specific growth rate at any time during the process. As shown in Figure 4, it is problematic to calculate the specific growth rate of early batches over the full duration of the batch, as lag-phases are prominently exhibited in the presented experimental data. Hence, calculating over the full duration instead of the exponential growth phase only will lead to considerably lower specific growth rates. This could explain the deviation between the lower specific growth rates in the early passages reported in the literature and the higher specific growth rates of early batches in the results presented in this work [25,26]. To avoid this, the authors propose using the specific growth rate during the exponential phase only and disregard the behavior during the lag phase for specific growth rate determination.

The CCD has been proposed in the literature as a time scale for measuring evolutionary change in ALE experiments [25]. However, it has been shown that ALE experiments in L scale bioreactors require a CCD that is two orders of magnitude larger than required on the mL scale to yield the same stable growth phenotype. Comparing the automated L scale system with the manual serial passaging approach, it is found that the manual approach requires considerably more passaging events than the automated repeated batch process to yield the same CCD. Thus, although the shake flask process itself is much slower regarding the total experimental time, it is more efficient when considering the number of propagated beneficial mutations per CCD. The optimization target, the heightened stable specific growth rate, depends on the frequency of beneficial mutations and thus the mutation rate in general. Assuming CCD as a transferable measure of evolutionary change between reactor scales, the assumption that the mutation rate behaves similar over time for the same experiment can be inferred. Hence, either the behavior of the mutation rate differs between the compared experimental setups or the deviation between the reported CCDs in the literature and the respective CCDs presented in this work must have another origin.

Campos et al. (2008) showed in their statistical analyses that population bottlenecks influence the population adaption rate [36]. Hence, the starting population size, the imposed bottleneck between batches, has an impact on the speed of evolution. LaCroix et al. (2017) investigated the relationship of the passage size on the adaption rate of the same biological system of *E. coli* K12 MG1655 growing on glycerol and found that increasing and constant passage sizes (0.001–10%) lead to a faster adaption regarding the specific growth rate [16]. In manual experiments, the initial biomass concentration is adjusted for each subsequent passage; hence, passage sizes vary and decrease over time. In the presented experiment, as well as in LaCroix et al. (2017), initial biomass concentrations between batches were kept constant, which contributes to the reduction in the experiment duration.

Mutation rate dynamics in bacterial populations are complex and some sources suggest that mutation rates can increase dramatically in stressful, changing environments in order to counterbalance robustness with evolvability [37–39]. Populations in uncontrolled shake flask experiments may be subjected to pH-shift and oxygen limitations. Furthermore, the passaging process as well as a potentially prolonged residence in the stationary growth

phase under limited substrate conditions may contribute to heightened stress levels and thus lead to elevated mutation rates and the lower cumulative number of cell divisions required in shake flask experiments. Thus, the controlled process conditions and the proposed drain/fill approach reduce the impact of unwanted selection pressures, including unnoticed fluctuations in process conditions or irregular and lengthy passaging procedures.

An automated adaptive laboratory evolution approach reduces the time required to perform ALE experiments in the presented use case. The time savings can be attributed to multiple sources. The efficient process timing reduces batch phases, which are traditionally spread out artificially over the course of 24 h, to the real time required until the end of the exponential growth phase, instead of prolonging this step by step-by-step reduction in the inoculum size. Consistently larger inoculum sizes result in more cell divisions per unit of time and a reduction in the lag phases [22–24]. The automated approach also reduces the time of passaging to approximately 42 min and reduces the amount of manual labor and at-line measurements.

Long-term cultivations can result in biofilm formation. In the context of ALE experiments, biofilm formation would result in an unwanted carry-over of biomass from previous batches. This may favor mutations that contribute to biofilm formation and thus introduce a new selection pressure. The extent of this stress decreases with an increasing reactor size as the relative fraction of passaged biofilm decreases with larger passage size volumes. In the presented use case of the repeated batch process, no visible biofilm formation was observed over the course of 200 h of cultivation. However, further extended cultivation times may eventually result in biofilm formation, especially if heavy frothing is encountered. The use of other microorganisms may result in issues with biofilm formation most likely along the stirrer axis or along the liquid level border. Anti-foam agents could be used to counter biofilm formation and reduce the biofilm build-up.

Many evolution experiments are run for a longer duration than the presented use-case. The maximum duration of the presented method is only dependent on the wear and tear of the bioreactor system. Maintenance during operation is restricted to the exchange of the off-gas air filters every four days to avoid potential pressure build up due to clogging

The current experimental set-up was limited to four parallel stirred-tank bioreactors. However, future improvements of this set-up may include miniaturization and further parallelization. Recent studies have shown the reproducibility and scalability of the used L scale stirred-tank bioreactor to a 48-parallelized-bioreactor system on the mL scale that is integrated into a liquid handling station [40]. With sophisticated bioreactor control software, such a set-up would improve the throughput of ALE experiments greatly and enable automated sampling and at-line sample anaylsis [41,42].

5. Conclusions

The authors propose a novel automated adaptive laboratory evolution method that transforms the conventional serial passaging approach from shake flask experiments into an L scale stirred-tank bioreactor that performs serial passaging, in the form sequential batches, automatically. The digitized experimental setup acquires on-line data of all key process and state variables that enable the detection of the end of the exponential growth phase and the online estimation of growth rates using a soft sensor approach. Automated ALE experiments in parallelized stirred-tank bioreactors enable the study of varying mutation rates within one experimental set-up at the same time. This novel ALE method can replace serial passaging experiments to speed up adaptive evolution experiments considerably. With the proposed method, the authors were able to obtain the same stable growth rates of *E. coli* K12 MG1655 with the non-native carbon source glycerol 9.4 times faster than reference serial passaging shake flask experiments and 3.6 times faster than an automated approach on the mL scale. It was further shown that the cumulative number of cell divisions (CCD) is not directly translatable between reactor scales. The cumulative number of cell divisions (CCD) required to obtain the same growth phenotypes as the reference process showed an offset of two orders of magnitude. It is concluded that the time necessary for

certain evolutionary progress is not only dependent on the CCD but also dependent on the scale and control of the experiment, as well as the number and size of passages.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/microorganisms11020275/s1.

Author Contributions: Conceptualization, L.B. and D.W.-B.; methodology, L.B.; software, L.B.; validation, L.B.; formal analysis, L.B.; investigation, L.B.; resources, D.W.-B.; data curation, L.B.; writing—original draft preparation, L.B.; writing—review and editing, D.W.-B.; visualization, L.B.; supervision, D.W.-B.; project administration, L.B. and D.W.-B.; funding acquisition, D.W.-B. All authors have read and agreed to the published version of the manuscript.

Funding: Funding was provided by the German Ministry of Education and Research (BMBF) within the national joint research project DigInBio [Grant number 031B0463B].

Data Availability Statement: The raw process data of the ALE experiments presented in this work, as well as a Jupyter Notebook that contains all data processing and analysis steps, are provided in the Supplementary Materials and on request.

Acknowledgments: Lukas Bromig thanks the support provided by the TUM Graduate School (Technical University of Munich, Germany). The authors also thank Nikolas von den Eichen (Chair of Biochemical Engineering, Technical University of Munich) and our project partners at Research Center Jülich and the University of Hannover for the helpful discussions and expertise on this topic. Lukas Bromig thanks the support provided by the TUM Graduate School (Technical University of Munich, Germany).

Conflicts of Interest: The authors declare no conflict of interest.

Sample Availability: Samples of the final stable phenotype strains are available upon request as cryogenically frozen stock-cultures.

Abbreviations

The following abbreviations are used in this manuscript:

ALE	Adaptive laboratory evolution
CCD	Cumulative number of cell divisions
CER	Carbon emission rate
DO	Dissolved oxygen
DoR	Degree of reduction
DCW	Dry cell weight
GR	Growth rate
LB	Lysogeny broth
NTG	N-methyl-N'nitro-N-nitrosoguanidine
OUR	Oxygen uptake rate
RB	Riesennberg
SiLA	Standardization in laboratory automation
WT	Wild type

References

1. Mavrommati, M.; Daskalaki, A.; Papanikolaou, S.; Aggelis, G. Adaptive laboratory evolution principles and applications in industrial biotechnology. *Biotechnol. Adv.* **2022**, *54*, 107795. [CrossRef] [PubMed]
2. Herring, C.D.; Raghunathan, A.; Honisch, C.; Patel, T.; Applebee, M.K.; Joyce, A.R.; Albert, T.J.; Blattner, F.R.; Van den Boom, D.; Cantor, C.R. Comparative genome sequencing of Escherichia coli allows observation of bacterial evolution on a laboratory timescale. *Nat. Genet.* **2006**, *38*, 1406–1412. [CrossRef]
3. Hua, Q.; Joyce, A.R.; Fong, S.S.; Palsson, B.Ø. Metabolic analysis of adaptive evolution for in silico-designed lactate-producing strains. *Biotechnol. Bioeng.* **2006**, *95*, 992–1002. [CrossRef]
4. Stanek, M.T.; Cooper, T.F.; Lenski, R.E. Identification and dynamics of a beneficial mutation in a long-term evolution experiment with Escherichia coli. *BMC Evol. Biol.* **2009**, *9*, 1–13. [CrossRef]
5. Conrad, T.M.; Lewis, N.E.; Palsson, B.Ø. Microbial laboratory evolution in the era of genome-scale science. *Mol. Syst. Biol.* **2011**, *7*, 509. [CrossRef] [PubMed]

6. Portnoy, V.A.; Bezdan, D.; Zengler, K. Adaptive laboratory evolution—harnessing the power of biology for metabolic engineering. *Curr. Opin. Biotechnol.* **2011**, *22*, 590–594. [CrossRef] [PubMed]
7. Dragosits, M.; Mattanovich, D. Adaptive laboratory evolution – principles and applications for biotechnology. *Microb. Cell Factories* **2013**, *12*, 64. [CrossRef]
8. Sandberg, T.E.; Salazar, M.J.; Weng, L.L.; Palsson, B.Ø.; Feist, A.M. The emergence of adaptive laboratory evolution as an efficient tool for biological discovery and industrial biotechnology. *Metab. Eng.* **2019**, *56*, 1–16. [CrossRef] [PubMed]
9. Stella, R.G.; Wiechert, J.; Noack, S.; Frunzke, J. Evolutionary engineering of Corynebacterium glutamicum. *Biotechnol. J.* **2019**, *14*, 1800444. [CrossRef]
10. Radek, A.; Tenhaef, N.; Müller, M.F.; Brüsseler, C.; Wiechert, W.; Marienhagen, J.; Polen, T.; Noack, S. Miniaturized and automated adaptive laboratory evolution: Evolving Corynebacterium glutamicum towards an improved d-xylose utilization. *Bioresour. Technol.* **2017**, *245*, 1377–1385. [CrossRef]
11. Si, T.; Chao, R.; Min, Y.; Wu, Y.; Ren, W.; Zhao, H. Automated multiplex genome-scale engineering in yeast. *Nat. Commun.* **2017**, *8*, 15187. [CrossRef] [PubMed]
12. Wang, J.; Jian, X.; Xing, X.H.; Zhang, C.; Fei, Q. Empowering a Methanol-Dependent Escherichia coli via Adaptive Evolution Using a High-Throughput Microbial Microdroplet Culture System. *Front. Bioeng. Biotechnol.* **2020**, *8*. [CrossRef] [PubMed]
13. Jian, X.; Guo, X.; Wang, J.; Tan, Z.L.; Xing, X.h.; Wang, L.; Zhang, C. Microbial microdroplet culture system (MMC): An integrated platform for automated, high-throughput microbial cultivation and adaptive evolution. *Biotechnol. Bioeng.* **2020**, *117*, 1724–1737. [CrossRef] [PubMed]
14. Wong, B.G.; Mancuso, C.P.; Kiriakov, S.; Bashor, C.J.; Khalil, A.S. Precise, automated control of conditions for high-throughput growth of yeast and bacteria with eVOLVER. *Nat. Biotechnol.* **2018**, *36*, 614–623. [CrossRef]
15. Ekkers, D.M.; Branco dos Santos, F.; Mallon, C.A.; Bruggeman, F.; van Doorn, G.S. The omnistat: A flexible continuous-culture system for prolonged experimental evolution. *Methods Ecol. Evol.* **2020**, *11*, 932–942. [CrossRef]
16. LaCroix, R.A.; Palsson, B.Ø.; Feist, A.M. A Model for Designing Adaptive Laboratory Evolution Experiments. *Appl. Environ. Microbiol.* **2017**, *83*, e03115-16. [CrossRef]
17. Ji, X.J.; Huang, H.; Ouyang, P.K. Microbial 2,3-butanediol production: A state-of-the-art review. *Biotechnol. Adv.* **2011**, *29*, 351–364. [CrossRef]
18. LaCroix, R.A.; Sandberg, T.E.; O'Brien, E.J.; Utrilla, J.; Ebrahim, A.; Guzman, G.I.; Szubin, R.; Palsson, B.Ø.; Feist, A.M. Use of Adaptive Laboratory Evolution To Discover Key Mutations Enabling Rapid Growth of Escherichia coli K-12 MG1655 on Glucose Minimal Medium. *Appl. Environ. Microbiol.* **2015**, *81*, 17–30. [CrossRef]
19. Wiser, M.J.; Lenski, R.E. A comparison of methods to measure fitness in Escherichia coli. *PLoS ONE* **2015**, *10*, e0126210. [CrossRef]
20. Bacun-Druzina, V.; Cagalj, Z.; Gjuracic, K. The growth advantage in stationary-phase (GASP) phenomenon in mixed cultures of enterobacteria. *FEMS Microbiol. Lett.* **2007**, *266*, 119–127. [CrossRef]
21. Sandberg, T.E.; Pedersen, M.; LaCroix, R.A.; Ebrahim, A.; Bonde, M.; Herrgard, M.J.; Palsson, B.Ø.; Sommer, M.; Feist, A.M. Evolution of Escherichia coli to 42 °C and Subsequent Genetic Engineering Reveals Adaptive Mechanisms and Novel Mutations. *Mol. Biol. Evol.* **2014**, *31*, 2647–2662. [CrossRef] [PubMed]
22. Bertrand, R.L. Lag Phase Is a Dynamic, Organized, Adaptive, and Evolvable Period That Prepares Bacteria for Cell Division. *J. Bacteriol.* **2019**, *201*, e00697-18. [CrossRef] [PubMed]
23. Augustin, J.C.; Brouillaud-Delattre, A.; Rosso, L.; Carlier, V. Significance of inoculum size in the lag time of Listeria monocytogenes. *Appl. Environ. Microbiol.* **2000**, *66*, 1706–1710. [CrossRef]
24. Robinson, T.P.; Aboaba, O.O.; Kaloti, A.; Ocio, M.J.; Baranyi, J.; Mackey, B.M. The effect of inoculum size on the lag phase of Listeria monocytogenes. *Int. J. Food Microbiol.* **2001**, *70*, 163–173. [CrossRef]
25. Lee, D.H.; Feist, A.M.; Barrett, C.L.; Palsson, B.Ø. Cumulative number of cell divisions as a meaningful timescale for adaptive laboratory evolution of Escherichia coli. *PLoS ONE* **2011**, *6*, e26172. [CrossRef]
26. Fong, S.S.; Joyce, A.R.; Palsson, B.Ø. Parallel adaptive evolution cultures of Escherichia coli lead to convergent growth phenotypes with different gene expression states. *Genome Res.* **2005**, *15*, 1365–1372. [CrossRef]
27. Riesenberg, D.; Schulz, V.; Knorre, W.A.; Pohl, H.D.; Korz, D.; Sanders, E.A.; Ross, A.; Deckwer, W.D. High cell density cultivation of Escherichia coli at controlled specific growth rate. *J. Biotechnol.* **1991**, *20*, 17–27. [CrossRef] [PubMed]
28. Kangwa, M.; Yelemane, V.; Polat, A.N.; Gorrepati, K.D.D.; Grasselli, M.; Fernández-Lahore, M. High-level fed-batch fermentative expression of an engineered Staphylococcal protein A based ligand in *E. coli*: Purification and characterization. *AMB Express* **2015**, *5*, 70. [CrossRef] [PubMed]
29. Bromig, L.; Leiter, D.; Mardale, A.V.; von den Eichen, N.; Bieringer, E.; Weuster-Botz, D. The SiLA 2 Manager for rapid device integration and workflow automation. *SoftwareX* **2022**, *17*, 100991. [CrossRef]
30. SiLA 2 Community. Sila 2 python reference implementation.
31. Roels, J.A. Application of macroscopic principles to microbial metabolism. *Biotechnol. Bioeng.* **1980**, *22*, 2457–2514. [CrossRef]
32. Villadsen, J.; Nielsen, J.; Lidén, G. Elemental and Redox Balances. In *Bioreaction Engineering Principles*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 63–118. [CrossRef]
33. Neidhardt, F.C. *Escherichia Coli and Salmonella: Typhimurium Cellular and Molecular Biology*; American Society for Microbiology: Washington, DC, USA, 1987. [CrossRef]

34. Herwig, C.; Marison, I.; Von Stockar, U. On-line stoichiometry and identification of metabolic state under dynamic process conditions. *Biotechnol. Bioeng.* **2001**, *75*, 345–354. [CrossRef] [PubMed]
35. Sagmeister, P.; Wechselberger, P.; Jazini, M.; Meitz, A.; Langemann, T.; Herwig, C. Soft sensor assisted dynamic bioprocess control: Efficient tools for bioprocess development. *Chem. Eng. Sci.* **2013**, *96*, 190–198. [CrossRef]
36. Campos, P.R.A.; Wahl, L.M. The Effects of Population Bottlenecks on Clonal Interference, and the Adaptation Effective Population Size. *Evolution* **2009**, *63*, 950–958. [CrossRef]
37. Wielgoss, S.; Barrick, J.E.; Tenaillon, O.; Wiser, M.J.; Dittmar, W.J.; Cruveiller, S.; Chane-Woon-Ming, B.; Médigue, C.; Lenski, R.E.; Schneider, D. Mutation rate dynamics in a bacterial population reflect tension between adaptation and genetic load. *Proc. Natl. Acad. Sci. USA* **2013**, *110*, 222–227. [CrossRef]
38. Swings, T.; Van den Bergh, B.; Wuyts, S.; Oeyen, E.; Voordeckers, K.; Verstrepen, K.J.; Fauvart, M.; Verstraeten, N.; Michiels, J. Adaptive tuning of mutation rates allows fast response to lethal stress in Escherichia coli. *Elife* **2017**, *6*, e22939. [CrossRef]
39. Sprouffske, K.; Aguilar-Rodríguez, J.; Sniegowski, P.; Wagner, A. High mutation rates limit evolutionary adaptation in Escherichia coli. *PLoS Genet.* **2018**, *14*, e1007324. [CrossRef]
40. Von den Eichen, N.; Bromig, L.; Sidarava, V.; Marienberg, H.; Weuster-Botz, D. Automated multi-scale cascade of parallel stirred-tank bioreactors for fast protein expression studies. *J. Biotechnol.* **2021**, *332*, 103–113. [CrossRef]
41. Bromig, L.; von den Eichen, N.; Weuster-Botz, D. Control of parallelized bioreactors I: Dynamic scheduling software for efficient bioprocess management in high-throughput systems. *Bioprocess Biosyst. Eng.* **2022**, *45*, 1927–1937. [CrossRef] [PubMed]
42. Osthege, M.; Dölle, M.; Bromig, L.; Wiechert, W.; Oldiges, M.; Weuster-Botz, D. Control of parallelized bioreactors II: Probabilistic quantification of carboxylic acid reductase activity for bioprocess optimization. *Bioprocess Biosyst. Eng.* **2022**, *45*, 1939–1954. [CrossRef]

7 Discussion and outlook

Research and development processes are dynamic and novel by nature. In this fast-paced environment, software solutions and architectures must be modular to enable quick iterations at low cost. The current software landscape of vertically integrated, non-interoperable solutions poses a barrier to sustainable digitalization. This cumulative dissertation addresses this challenge by presenting a series of works that collectively establish a reusable, standards-based integration platform for laboratory automation. The foundation was laid with the development of the SiLA 2 Manager [Bromig *et al.* 2022a], a microservice-based software architecture that enables modular and decoupled communication with laboratory devices through SiLA 2-compliant interfaces. This infrastructure separates application-specific logic from device integration, allowing for independent development and reuse.

Building on this, the same integration layer was reused and extended in Bromig *et al.* 2022b to support additional instruments and use cases. The publication introduced a dynamic scheduling application for orchestrating liquid handling operations for the process control of an integrated parallelized bioreactor system – demonstrating how new experimental logic can be deployed on top of the existing architecture without modifying the underlying integration platform. This further validated the system's modularity and flexibility.

In Bromig *et al.* 2023, the platform was applied to a more complex bioprocessing scenario: automated adaptive laboratory evolution. The SiLA 2 Manager and previously developed connectors – such as those for the L-scale parallel bioreactor system, off-gas analysis, and peristaltic pumps – were reused to implement real-time monitoring and control, as well as biomass estimation via a soft sensor implementation. This highlights

the scalability of the architecture and the benefits of using open standards for enabling reproducible and extensible automation workflows in industrial biotechnology.

Taken together, these works demonstrate a modular, distributed, and modern architecture for laboratory integration. Leveraging interface standards like SiLA 2 and open source frameworks, the proposed platform overcomes the rigidity of vertically integrated solutions and paves the way for agile, scalable laboratory automation. While integration platforms are well established in other industries – and to a limited extent in the life sciences – they have traditionally focused only on data flow. Looking forward, these systems must be extended or replaced by architectures that also support control flow, enabling low-level, bidirectional integration required by modern use cases in autonomous experimentation.

The architecture and integration strategy demonstrated in this dissertation aligns with broader trends and conceptual frameworks in industrial automation research. Biermann *et al.* 2021 highlight the challenges posed by the lack of standardization and the fragmented software landscape in biotechnological laboratories, reinforcing the need for modular and interoperable integration platforms like the SiLA 2 Manager. Their discussion of middleware architectures with device-specific adapters and the importance of flexible, vendor-agnostic communication interfaces validates the microservice approach adopted in this work. Similarly, Pallasch *et al.* 2018 propose a layered control architecture in which edge-level devices act as operational containers, abstracting hardware and enabling cloud-assisted control. The SiLA 2 Manager reflects a comparable architectural decoupling, supporting modular software deployment and reusable device connectors while maintaining real-time capabilities. Although the presented implementation remains locally scoped, the infrastructure could be extended to support cloud-assisted orchestration, versioned control logic, and even integration with semantic device descriptions as proposed through Asset Administration Shells (AAS).

These parallels highlight the forward compatibility of the presented system with ongoing developments in both industrial automation and laboratory digitization.

The lack of standardized device interfaces leads to a high cost of digitalization and automation projects. While SiLA 2 is steadily gaining ground internationally as the leading interface standard, adoption remains slow. The upcoming companion specification of the established OPC UA standard, LADS, released and verified by the OPC Foundation in January 2024, promises additional awareness and standard adoption in the laboratory domain in the years to come. As SiLA 2 has a larger community and is simpler to learn and develop with, it is particularly well suited for end-users, academics, and solution providers to retro-fit non-standardized device interfaces. OPC UA LADS is more welcomed by hardware vendors, as it requires more specialist knowledge, device information, and deeper access to implement properly.

With only small relevant technical differences, it is assumed that both standards will co-exist in the future. However, OPC UA already offers a self-descriptive ontology which – in case this is not developed for the SiLA 2 standard - may prove as a major deciding factor in favor of OPC UA LADS adoption, especially if AI applications in the laboratory automation environment become more common-place. A coexistence of both standards is yet another reason in favor of integration platforms as they can provide the respective mappings between both standards.

The laboratory of the future is not just designed for human researchers, but for machines and software. An integration platform is a first step towards this, as it provides machine-readable and executable access to the laboratory layer. Applying this to more concrete examples in bioprocessing could speed up scientific discovery. For example, Bayesian optimization (BO) is becoming increasingly used for tasks ranging from assay development to bioprocess optimization and media development [Gisperg *et al.* 2024; Helleckes *et al.* 2023a,b]. BO is a promising candidate because its iterative

nature makes it a suitable candidate for a closed-loop automation.

Current trends in lab automation research focuses on the development of an AI scientist and the self-driven lab. Research consortia, like the Acceleration Consortium in Canada or Future House, a private non-profit organization, are evaluating large language models (LLMs) for their applicability in the laboratory environment [Future House Inc. 2024; Yoshikawa *et al.* 2023] with the goal of fully automating laboratory research with AI agents. For this, a homogeneous programmatic and self-descriptive interface is required to provide respective AI agents with the capability to execute. While currently relying on pseudo-code with no link to the device interfaces [O'Donoghue *et al.* 2023; Yoshikawa *et al.* 2023], an integration platform, as suggested in this work, would close this gap. Looking forward, future developments should include these requirements as well as requirements stemming from other industry consortia that foster interoperability and overarching connectivity, such as Industrie 4.0 and its specification for the Reference Architecture Model Industrie 4.0 (RAMI 4.0) and the Administration Asset Shell (AAS).

Data requires context to become information. This also holds true for system interfaces. While standards like SiLA 2 and OPC UA provide a self-descriptive interface, a lot of contextual and meta data about the devices, processes, and result data is missing. To achieve fully autonomous, self-driven laboratories, this contextual and meta data is required in a machine readable format. While the FAIR data concept describes this from the problem, top-down perspective, it provides little to no actionable solutions or guidance for its implementation. A semantic model of the laboratory domain is required, that combines various ontologies from labware to lab processes and methodologies, to devices and types of data. Bai *et al.* 2022 mapped the current landscape of available models, schemata, and databases for this purpose [Bai *et al.* 2022], but further work must be directed to develop a full working semantic model of the laboratory.

Concrete examples for bioprocessing include the development of device ontologies encompassing bioreactors, periphery devices, and analytical instruments, as well as data schemata for the respective device data types. Similar projects have been realized for material science laboratory equipment and are currently in progress under the umbrella of the NFDI4CAT program funded by the German Research Foundation (DFG, Project number 441926934) [DECHEMA and NFDI4Cat Consortium 2025; Jalali *et al.* 2023]. In combination with standardized device interfaces, this would not just accelerate the development and implementation of modern applications, like soft-sensors and advanced bioreactor control systems, but also ensure that resulting data is sufficiently enriched and annotated for data sharing and training of AI models and advanced analyses.

The presented architecture offers inherent scalability that enables its deployment beyond single-lab environments, making it suitable for larger laboratory infrastructures, multi-site research facilities, and cloud-based automation paradigms such as biofoundries or cloud labs. Its modular design and use of open standards like SiLA 2 facilitate reuse and replication across different settings; however, transitioning the presented software architecture to industrial use – such as in biotech companies, contract research organizations (CROs), or contract development and manufacturing organizations (CDMOs) – requires substantial further development. This includes rigorous validation, expansion of supported devices in the form of SiLA 2-compliant connector applications, and the implementation of features necessary to meet regulatory demands. At its current maturity level, the SiLA 2 Manager platform remains a research-focused prototype. Moreover, integration with existing LIMS, LES, or ELN systems is a further future step. Such integration would align with the overarching goal of reducing the number of user-facing interfaces in the laboratory, allowing researchers and operators to work within a unified digital environment while the automation infrastructure operates transparently in the background. This interoperability would be key to bridging

the gap between flexible academic systems and the structured workflows demanded in industrial settings.

Looking ahead, the convergence of modular integration platforms, standardized device communication, and semantic interoperability frameworks represents a critical enabler for the autonomous and intelligent laboratory. The architecture presented in this work lays a foundation for such developments by demonstrating how open standards, microservice-based design, and reusable software components can support flexible and scalable automation. As laboratory environments become increasingly data-driven and AI-assisted, the need for programmable, self-descriptive, and interoperable infrastructures will only intensify. Bridging laboratory automation with cloud infrastructure, advanced analytics, and digital twin technologies will be essential for realizing the full potential of autonomous research environments. Continued interdisciplinary collaboration – spanning software engineering, biotechnology, systems integration, and science – will be key to transitioning from research prototypes to robust, production-ready software that can drive innovation in both academic and industrial settings.

8 Bibliography

1. Abdou, M., Ezz, A. M. & Farag, I. *Digital Automation Platforms Comparative Study* in *2021 4th International Conference on Information and Computer Technologies (ICICT)* (Mar. 2021), 279–286.

2. Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., Bodenstein, S. W., Evans, D. A., Hung, C.-C., O'Neill, M., Reiman, D., Tunyasuvunakool, K., Wu, Z., Žemgulytė, A., Arvaniti, E., Beattie, C., Bertolli, O., Bridgland, A., Cherepanov, A., Congreve, M., Cowen-Rivers, A. I., Cowie, A., Figurnov, M., Fuchs, F. B., Gladman, H., Jain, R., Khan, Y. A., Low, C. M. R., Perlin, K., Potapenko, A., Savy, P., Singh, S., Stecula, A., Thillaisundaram, A., Tong, C., Yakneen, S., Zhong, E. D., Zielinski, M., Žídek, A., Bapst, V., Kohli, P., Jaderberg, M., Hassabis, D. & Jumper, J. M. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature* **630,** 493–500 (2024).

3. Allotrope Foundation. *ASM* `https://gitlab.com/allotrope-public/asm`. 2023.

4. Arm, J., Benesl, T., Marcon, P., Bradac, Z., Schröder, T., Belyaev, A., Werner, T., Braun, V., Kamensky, P., Zezulka, F., Diedrich, C. & Dohnal, P. Automated Design and Integration of Asset Administration Shells in Components of Industry 4.0. *Sensors (Basel, Switzerland)* **21,** 2004 (Mar. 2021).

5. Arnold, C. Cloud labs: where robots do the research. en. *Nature* **606,** 612–613 (June 2022).

6. Bai, J., Cao, L., Mosbach, S., Akroyd, J., Lapkin, A. A. & Kraft, M. From Platform to Knowledge Graph: Evolution of Laboratory Automation. *JACS Au* **2,** 292–309 (Feb. 2022).

7. Bär, H., Hochstrasser, R. & Papenfuß, B. SiLA: Basic Standards for Rapid Integration in Laboratory Automation. en. *Journal of Laboratory Automation* **17,** 86–95 (Apr. 2012).

8. Bartley, B., Beal, J., Rogers, M., Bryce, D., Goldman, R. P., Keller, B., Lee, P., Biggers, V., Nowak, J. & Weston, M. Building an Open Representation for Biological Protocols. *J. Emerg. Technol. Comput. Syst.* **19,** 28:1–28:21 (June 2023).

9. Benner, P. *Miniaturisierte Rührreaktoren mit LED-Beleuchtung zur photoautotrophen Hochdurchsatz-Bioprozessentwicklung* de. PhD thesis (Technische Universität München, 2024), 141.

10. Benner, P., Lüdtke, F. J., Beyer, N., von den Eichen, N., Oropeza Vargas, J. E. & Weuster-Botz, D. LED illumination modules enable automated photoautotrophic cultivation of Microalgae in parallel milliliter-scale stirred-Tank Bioreactors. *Applied Sciences* **13,** 5064 (2023).

11. Beutel, S. & Lenk, F. *Smart Biolabs of the Future* en (eds Beutel, S. & Lenk, F.) (Springer International Publishing, Cham, 2022).

12. Biermann, F., Mathews, J., Nießing, B., König, N. & Schmitt, R. H. Automating Laboratory Processes by Connecting Biotech and Robotic Devices—An Overview of the Current Challenges, Existing Solutions and Ongoing Developments. en. *Processes* **9,** 966 (June 2021).

13. Bloomberg, J. *Cloud-Centered IT Transformation At AstraZeneca* en. 2016.

14. Blums, K., Herzog, J., Costa, J., Quirico, L., Turber, J. & Weuster-Botz, D. Automation of RNA-Seq Sample Preparation and Miniaturized Parallel Bioreactors Enable High-Throughput Differential Gene Expression Studies. en. *Microorganisms* **13,** 849 (Apr. 2025).

15. Brendel, A., Dorfmüller, F., Liebscher, A., Kraus, P., Kress, K., Oehme, H., Arnold, M. & Koschitzki, R. en. in *Smart Biolabs of the Future* 175–194 (Cham, 2022).

16. Bromig, L., Leiter, D., Mardale, A.-V., von den Eichen, N., Bieringer, E. & Weuster-Botz, D. The SiLA 2 Manager for rapid device integration and workflow automation. *SoftwareX* **17,** 100991 (2022).

17. Bromig, L., Von den Eichen, N. & Weuster-Botz, D. Control of parallelized bioreactors I: dynamic scheduling software for efficient bioprocess management in high-throughput systems. en. *Bioprocess and Biosystems Engineering* **45,** 1927–1937 (2022).

18. Bromig, L. & Weuster-Botz, D. Accelerated adaptive laboratory evolution by automated repeated batch processes in parallelized bioreactors. *Microorganisms* **11,** 275 (2023).

19. Chen, A., Zhang, B. & Bao, J. Adaptive evolution of *Paecilomyces variotii* enhanced the biodetoxification of high-titer inhibitors in pretreated lignocellulosic feedstock. *Bioresource Technology* **411,** 131351 (2024).

20. DECHEMA and NFDI4Cat Consortium. *NFDI4Cat* en-US.

21. Della Corte, D., Colsman, W., Fessenmayr, H., Sawczuk da Silva, A. & Vanderwall, D. E. Self-reporting data assets and their representation in the pharmaceutical industry. *Drug Discovery Today* **27,** 207–214 (Jan. 2022).

22. Domínguez, J. A., Fuentes, R. P., Vegetti, M., Roldán, L., Gonnet, S. & Diván, M. J. *Ontology Implementation of OPC UA and AutomationML: A Review* en. in *Advanced Intelligent Technologies for Industry* (Singapore, 2022), 17–26.

23. Ehie, I. C. & Chilton, M. A. Understanding the influence of IT/OT Convergence on the adoption of Internet of Things (IoT) in manufacturing organizations: An empirical investigation. en. *Computers in Industry* **115,** 103166 (Feb. 2020).

24. For Research and Innovation (European Commission), D.-G. & Services, P. E. *Cost-benefit analysis for FAIR research data: cost of not having FAIR research data* eng (Publications Office of the European Union, LU, 2018).

25. Freundel, M. en. in *Smart Biolabs of the Future* 133–145 (Cham, 2022).

26. Future House Inc. *FutureHouse*

27. Gauglitz, G. Lab 4.0: SiLA or OPC UA. en. *Analytical and Bioanalytical Chemistry* **410,** 5093–5094 (Aug. 2018).

28. Giannelli, C. & Picone, M. Editorial "Industrial IoT as IT and OT Convergence: Challenges and Opportunities". en. *IoT* **3,** 259–261 (Mar. 2022).

29. Gisperg, F., Klausser, R., Elshazly, M., Kopp, J., Brichtová, E. P. & Spadiut, O. *Bayesian Optimization in Bioprocess Engineering -Where do we stand today?* 2024.

30. Habich, T. & Beutel, S. Digitalization concepts in academic bioprocess development. eng. *Engineering in Life Sciences* **24,** 2300238 (Apr. 2024).

31. Halle, L., Hollmann, N., Tenhaef, N., Mbengi, L., Glitz, C., Wiechert, W., Polen, T., Baumgart, M., Bott, M. & Noack, S. Robotic workflows for automated long-term adaptive laboratory evolution: improving ethanol utilization by Corynebacterium glutamicum. en. *Microbial Cell Factories* **22,** 175 (2023).

32. Helleckes, L. M., Hemmerich, J., Wiechert, W., Lieres, E. v. & Grünberger, A. Machine learning in bioprocess development: from promise to practice. English. *Trends in Biotechnology* **41,** 817–835 (June 2023).

33. Helleckes, L. M., Müller, C., Griesbach, T., Waffenschmidt, V., Moch, M., Osthege, M., Wiechert, W. & Oldiges, M. Explore or exploit? A model-based screening strategy for PETase secretion by Corynebacterium glutamicum. en. *Biotechnology and Bioengineering* **120,** 139–153 (2023).

34. Hildebrandt, C., Köcher, A., Küstner, C., López-Enríquez, C.-M., Müller, A. W., Caesar, B., Gundlach, C. S. & Fay, A. Ontology Building for Cyber–Physical Systems: Application in the Manufacturing Domain. *IEEE Transactions on Automation Science and Engineering* **17,** 1266–1282 (July 2020).

35. Hinkel, G., Kunert, J. & Meredith, J. The Tecan SiLA2 SDK: A royalty-free, open-source framework to develop SiLA2 servers and clients. *SLAS Technology* **28,** 334–344 (Oct. 2023).

36. Hohmann, S. en. in *Digital Transformation of the Laboratory* 143–147 (2021).

37. Holland, I. & Davies, J. A. Automation in the Life Science Research Laboratory. *Frontiers in Bioengineering and Biotechnology* **8,** 571777 (Nov. 2020).

38. ISPE | International Society for Pharmaceutical Engineering. *GAMP 5 Guide 2nd Edition* en. 2022.

39. Jalali, M., Mail, M., Aversa, R. & Kübel, C. MSLE: An ontology for materials science laboratory equipment – Large-scale devices for materials characterization. *Materials Today Communications* **35,** 105532 (June 2023).

40. Jessop-Fabre, M. M. & Sonnenschein, N. Improving Reproducibility in Synthetic Biology. English. *Frontiers in Bioengineering and Biotechnology* **7** (Feb. 2019).

41. Juchli, D. en. in *Smart Biolabs of the Future* 147–174 (Cham, 2022).

42. Kranjc, T. en. in *Digital Transformation of the Laboratory* 75–84 (2021).

43. Krüger, A., Schäfers, C., Busch, P. & Antranikian, G. Digitalization in microbiology – Paving the path to sustainable circular bioeconomy. en. *New Biotechnology* **59,** 88–96 (Nov. 2020).

44. Laurent, J. M., Janizek, J. D., Ruzo, M., Hinks, M. M., Hammerling, M. J., Narayanan, S., Ponnapati, M., White, A. D. & Rodriques, S. G. *LAB-Bench: Measuring Capabilities of Language Models for Biology Research* July 2024.

45. Logan, Z., Undieh, K. & Goli, M. *Autonomous Integration of Bench-Top Wet Lab Equipment* Aug. 2024.

46. Mavrommati, M., Daskalaki, A., Papanikolaou, S. & Aggelis, G. Adaptive laboratory evolution principles and applications in industrial biotechnology. en. *Biotechnology Advances* **54,** 107795 (2022).

47. Merz, K. M., Amaro, R., Cournia, Z., Rarey, M., Soares, T., Tropsha, A., Wahab, H. A. & Wang, R. Editorial: Method and Data Sharing and Reproducibility of Scientific Results. *Journal of Chemical Information and Modeling* **60,** 5868–5869 (Dec. 2020).

48. Mione, F. M., Kaspersetz, L., Luna, M. F., Aizpuru, J., Scholz, R., Borisyak, M., Kemmer, A., Schermeyer, M. T., Martinez, E. C., Neubauer, P. & Cruz Bournazou, M. N. A workflow management system for reproducible and interoperable high-throughput self-driving experiments. *Computers & Chemical Engineering* **187,** 108720 (Aug. 2024).

49. Miyakawa, T. No raw data, no science: another possible source of the reproducibility crisis. *Molecular Brain* **13,** 24 (Feb. 2020).

50. Neumann, J. en. in *Smart Biolabs of the Future* 195–207 (Cham, 2022).

51. O'Donoghue, O., Shtedritski, A., Ginger, J., Abboud, R., Ghareeb, A. E., Booth, J. & Rodriques, S. G. *BioPlanner: Automatic Evaluation of LLMs on Protocol Planning in Biology* Oct. 2023.

52. OPC Foundation. *LADS - Laboratory and Analytical Device Standard* en-US.

53. Orsi, E., Schada von Borzyskowski, L., Noack, S., Nikel, P. I. & Lindner, S. N. Automated in vivo enzyme engineering accelerates biocatalyst optimization. *Nature Communications* **15,** 3447 (2024).

54. Pallasch, C., Wein, S., Hoffmann, N., Obdenbusch, M., Buchner, T., Waltl, J. & Brecher, C. *Edge Powered Industrial Control: Concept for Combining Cloud and Automation Technologies* in *2018 IEEE International Conference on Edge Computing (EDGE)* (July 2018), 130–134.

55. Rathore, A. S., Nikita, S. & Jesubalan, N. G. Digitization in bioprocessing: The role of soft sensors in monitoring and control of downstream processing for production of biotherapeutic products. *Biosensors and Bioelectronics: X* **12,** 100263 (Dec. 2022).

56. Ravazzi, P. & Villa, A. en. in *Springer Handbook of Automation* 93–116 (Berlin, Heidelberg, 2009).

57. Ribeiro, B., Meckin, R., Balmer, A. & Shapira, P. The digitalisation paradox of everyday scientific labour: How mundane knowledge work is amplified and diversified in the biosciences. *Research Policy* **52,** 104607 (Jan. 2023).

58. Russell, K. en. in *Digital Transformation of the Laboratory* 101–105 (2021).

59. Sandberg, T. E., Salazar, M. J., Weng, L. L., Palsson, B. & Feist, A. M. The emergence of adaptive laboratory evolution as an efficient tool for biological discovery and industrial biotechnology. en. *Metabolic Engineering* **56,** 1–16 (2019).

60. Scheitz, C. J. F., Peck, L. J. & Groban, E. S. Biotechnology software in the digital age: are you winning? en. *Journal of Industrial Microbiology and Biotechnology* **45,** 529–534 (July 2018).

61. Seidel, S., Cruz-Bournazou, M. N., Groß, S., Schollmeyer, J. K., Kurreck, A., Krauss, S. & Neubauer, P. en. in *Smart Biolabs of the Future* 61–82 (Cham, 2022).

62. SiLA 2 Consortium. *SiLA 2 Part (B) - Mapping Specification* de.

63. SiLA Consortium. *SiLA 2 Part (A) - Overview, Concepts and Core Specification* de.

64. Sim, M., Vakili, M. G., Strieth-Kalthoff, F., Hao, H., Hickman, R. J., Miret, S., Pablo-García, S. & Aspuru-Guzik, A. ChemOS 2.0: An orchestration architecture for chemical self-driving laboratories. English. *Matter* **7,** 2959–2977 (Sept. 2024).

65. Skarlinski, M. D., Cox, S., Laurent, J. M., Braza, J. D., Hinks, M., Hammerling, M. J., Ponnapati, M., Rodriques, S. G. & White, A. D. *Language agents achieve superhuman synthesis of scientific knowledge* Sept. 2024.

66. Steindl, G. & Kastner, W. *Transforming OPC UA Information Models into Domain-Specific Ontologies* in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)* (May 2021), 191–196.

67. Steinwandter, V., Borchert, D. & Herwig, C. Data science tools and applications on the way to Pharma 4.0. en. *Drug Discovery Today* **24,** 1795–1805 (Sept. 2019).

68. Teytelman, L., Stoliartchouk, A., Kindler, L. & Hurwitz, B. L. Protocols.io: Virtual Communities for Protocol Development and Discussion. en. *PLOS Biology* **14,** e1002538 (Aug. 2016).

69. Tom, G., Schmid, S. P., Baird, S. G., Cao, Y., Darvish, K., Hao, H., Lo, S., Pablo-García, S., Rajaonson, E. M., Skreta, M., Yoshikawa, N., Corapi, S., Akkoc, G. D.,

Strieth-Kalthoff, F., Seifrid, M. & Aspuru-Guzik, A. Self-Driving Laboratories for Chemistry and Materials Science. *Chemical Reviews* **124,** 9633–9732 (Aug. 2024).

70. U.S. FDA. *21 CFR Part 11 – Electronic Records; Electronic Signatures* en. 2024.

71. US Congressional Budget Office. *Research and Development in the Pharmaceutical Industry* en. Tech. rep. 57025 (US Congressional Budget Office, Aug. 2021).

72. Von den Eichen, N., Bromig, L., Sidarava, V., Marienberg, H. & Weuster-Botz, D. Automated multi-scale cascade of parallel stirred-tank bioreactors for fast protein expression studies. *Journal of Biotechnology* **332,** 103–113 (2021).

73. Von den Eichen, N., Osthege, M., Dölle, M., Bromig, L., Wiechert, W., Oldiges, M. & Weuster-Botz, D. Control of parallelized bioreactors II: probabilistic quantification of carboxylic acid reductase activity for bioprocess optimization. en. *Bioprocess and Biosystems Engineering* **45,** 1939–1954 (2022).

74. Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., Gonzalez-Beltran, A., Gray, A. J., Groth, P., Goble, C., Grethe, J. S., Heringa, J., 'T Hoen, P. A., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S. J., Martone, M. E., Mons, A., Packer, A. L., Persson, B., Rocca-Serra, P., Roos, M., Van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M. A., Thompson, M., Van Der Lei, J., Van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J. & Mons, B. The FAIR Guiding Principles for scientific data management and stewardship. en. *Scientific Data* **3,** 160018 (Mar. 2016).

75. Wolf, F. A., Angerer, P. & Theis, F. J. SCANPY: large-scale single-cell gene expression data analysis. en. *Genome Biology* **19,** 15 (Feb. 2018).

76. Yoshikawa, N., Skreta, M., Darvish, K., Arellano-Rubach, S., Ji, Z., Bjørn Kristensen, L., Li, A. Z., Zhao, Y., Xu, H., Kuramshin, A., Aspuru-Guzik, A., Shkurti, F. & Garg, A. Large language models for chemistry robotics. en. *Autonomous Robots* **47,** 1057–1086 (Dec. 2023).

77. Zuchowski, R., Schito, S., Neuheuser, F., Menke, P., Berger, D., Hollmann, N., Gujar, S., Sundermeyer, L., Mack, C., Wirtz, A., Weiergräber, O. H., Polen, T., Bott, M., Noack, S. & Baumgart, M. Discovery of novel amino acid production traits by evolution of synthetic co-cultures. en. *Microbial Cell Factories* **22,** 71 (2023).

78. Zupancic, K., Pavlek, T. & Erjavec, J. *Digital Transformation of the Laboratory: A Practical Guide to the Connected Lab* (John Wiley & Sons, 2021).

Abbreviations

Abbreviation	Meaning
AAS	Administration Asset Shell
AFO	Allotrope Foundation Ontologies
AI	Artificial Intelligence
ALE	Adaptive Laboratory Evolution
AnIML	Analytical Information Markup Language
API	Application Programming Interface
ASM	Allotrope Simple Model
ASTM	American Society for Testing and Materials
BMBF	Federal Ministry of Education and Research
BO	Bayesian Optimization
CCD	Cumulative Number of Cell Divisions
CDMO	Contract Development and Manufacturing Organization
CRO	Contract Research Organization
CSV	Comma-separated values
DIY	Do It Yourself
ELN	Electronic Lab Notebook
ETL	Extract-Transform-Load
FAIR	Findable, Accessible, Interoperable, Reusable
FDA	Federal Drug Administration
F&E	Forschung und Entwicklung
GAMP	Good Automated Manufacturing Practice
GDPR	General Data Protection Regulation

gRPC	"g" Remote Procedure Call
GxP	Good Practice (x standing for various fields, e.g. manufacturing, laboratory)
HTTP	Hypertext Transfer Protocol
IDS	Intermediary Data Schemas
IIoT	Industrial Internet of Things
IoT	Internet of Things
ISPE	International Society for Pharmaceutical Engineering
IT	Information Technology
JSON	JavaScript Object Notation
LADS	Laboratory and Analytical Device Standard
LES	Lab Execution System
LIMS	Laboratory Information Management System
LLM	Large Language Model
MQTT	Message Queuing Telemetry Transport
NGS	Next-Generation Sequencing
OD	Optical Density
OPC UA	Open Platform Communication Unified Architecture
OT	Operational Technology
PAT	Process Analytical Technology
RAMI 4.0	Reference Architecture Model Industrie 4.0
R&D	Research and Development
REST	Representational State Transfer
SiLA	Standardization in Laboratory Automation
SOP	Standard Operating Procedure
XML	Extensible Markup Language