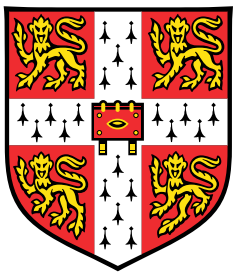
A dynamic knowledge graph approach to self-driving chemical laboratories



Jiaru Bai

Supervisor: Prof. Markus Kraft

Department of Chemical Engineering and Biotechnology
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Wolfson College

September 2023

I would like to dedicate this thesis to my loving parents and partner

行百里者半九十

《战国策·秦策五》

In a journey of a hundred *li* (miles), the first ninety *li* is merely half of it.

Stratagems of the Warring States

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.
Some of the work in this dissertation has been published, accepted for publication, or submitted for publication:

1. **Bai, J.**, Geeson, R., Farazi, F., Mosbach, S., Akroyd, J., Bringley, E.J. and Kraft, M. (2021). Automated Calibration of a Poly(Oxymethylene) Dimethyl Ether Oxidation Mechanism Using the Knowledge Graph Technology. Journal of Chemical Information and Modeling, 61(4):1701–1717. doi:10.1021/acs.jcim.0c01322.

2. **Bai, J.**, Cao, L., Mosbach, S., Akroyd, J., Lapkin, A.A. and Kraft, M. (2022). From Platform to Knowledge Graph: Evolution of Laboratory Automation. JACS Au, 2(2):292–309. doi:10.1021/jacsau.1c00438.

3. **Bai, J.**, Lee, K.F., Hofmeister, M., Mosbach, S., Akroyd, J. and Kraft, M. (2024). A Derived Information Framework for a Dynamic Knowledge Graph and its Application to Smart Cities. Future Generation Computer Systems, 152:112–126. doi:10.1016/j.future.2023.10.008.

4. **Bai, J.**, Mosbach, S., Taylor, C.J., Karan, D., Lee, K.F., Rihm, S.D., Akroyd, J., Lapkin, A.A. and Kraft, M. (2024). A Dynamic Knowledge Graph Approach to Distributed Self-Driving Laboratories. Nature Communications, 15:462. doi:10.1038/s41467-023-44599-9.

The work presented in this dissertation has also contributed, to a lesser degree, to the following (submitted) publications:

1. Tan, Y.R., Salamanca, M., **Bai, J.**, Akroyd, J. and Kraft, M. (2021). Structural Effects of C3 Oxygenated Fuels on Soot Formation in Ethylene Coflow Diffusion Flames. Combustion and Flame, 232:111512. doi:10.1016/j.combustflame.2021.111512.

2. Kondinski, A., **Bai, J.**, Mosbach, S., Akroyd, J. and Kraft, M. (2023). Knowledge Engineering in Chemistry: From Expert Systems to Agents of Creation. Accounts of Chemical Research, 56(2):128–139. doi:10.1021/acs.accounts.2c00617.

3. Hofmeister, M., **Bai, J.**, Brownbridge, G., Mosbach, S., Lee, K.F., Farazi, F., Hillman, H., Agarwal, M., Ganguly, S., Akroyd, J. and Kraft, M. (submitted). Semantic Agent Framework for Automated Flood Assessment Using Dynamic Knowledge Graphs.

4. Taylor, C.J., **Bai, J.**, Zawodny, W., Grainger, R., Felton, K.C., Mosbach, S., Kraft, M., Chessari, G., Johnson, C. and Lapkin, A.A. (in preparation). An Optimisation Workflow to Enable Challenging Synthetic Pathways.

Other works not presented in this thesis include:

1. Eibeck, A., Nurkowski, D., Menon, A., **Bai, J.**, Wu, J., Zhou, L., Mosbach, S., Akroyd, J. and Kraft, M. (2021). Predicting Power Conversion Efficiency of Organic Photovoltaics: Models and Data Analysis. ACS Omega, 6(37):23764–23775. doi:10.1021/acsomega.1c02156.

2. Rihm, S.D., **Bai, J.**, Pascazio, L. and Kraft, M. (2023). Fully Automated Kinetic Models Extend our Understanding of Complex Reaction Mechanisms. Chemie Ingenieur Technik, 95(5):740–748. doi:10.1002/cite.202200220.

3. Kondinski, A., Mosbach, S., Akroyd, J., Breeson, A., Tan, Y.R., Rihm, S.D., **Bai, J.** and Kraft, M. (2024). Hacking Decarbonization with a Community-Operated CreatorSpace. Chem, in press. doi:10.1016/j.chempr.2023.12.018.

4. Xie, W., Atherton, J., **Bai, J.**, Farazi, F., Mosbach, S., Akroyd, J. and Kraft, M. (submitted). A Nuclear Future? Small Modular Reactors in a Carbon Tax-Driven Transition to Clean Energy.

Jiaru Bai
September 2023

Acknowledgements

I would like to express my sincere thanks to my supervisor Prof. Markus Kraft for his supervision and support throughout my study. While some of his ideas may have seemed unconventional at first, they often turned out to be truly inspiring. I would also like to extend my gratitude to Dr Sebastian Mosbach for his advice and the many insightful discussions we have had over the years. His work ethic and extensive knowledge have constantly motivated me to tackle many challenging problems. Additionally, I wish to thank Dr Jethro Akroyd for his advice and the energy he brought to our interactions, which encouraged me to aim higher in my pursuits.

I want to thank the members of the Computational Modelling Group for their friendship and help: Feroz Farazi, Aleksandar Kondinski, Kok Foong Lee, Markus Hofmeister, Yong Ren Tan, Wanni Xie, Xiaochi Zhou, Simon D. Rihm, Angiras Menon, Rory M. Geeson, Eric J. Bringley, George Brownbridge, Andreas Eibeck, and Daniel Nurkowski.

I also want to thank the members of the Sustainable Reaction Engineering Group for the fruitful collaboration: Prof. Alexei Lapkin, Dr Connor Taylor, Dr Dogancan Karan, and Dr Liwei Cao.

I would like to thank my college tutor Dr Ana Toribio for her support throughout my PhD studies.

I am very grateful for the financial support provided by the CSC Cambridge International Scholarship from Cambridge Trust and China Scholarship Council.

I extend my heartfelt gratitude to my viva examiners Profs. Ross King and Lee Cronin for engaging in insightful discussions and providing valuable comments on this thesis.

Last, but most importantly, I thank my partner Yiqun for her unwavering support and for ensuring I was in great spirits throughout my high school, undergraduate and postgraduate years, and my parents Y.W. Bai and P.Q. Sun for their unconditional love and support. This thesis would not exist without you.

I also appreciate all the support I received from the rest of my family and my friends.

Abstract

The contemporary design of self-driving laboratories faces difficulties in scalability and interoperability when it comes to the vision of a globally connected research network. This is due to heterogeneous data formats and resources as an obstacle to holistic integration. This thesis investigates a potential solution to the interoperability problem in chemical experiments by utilising a dynamic knowledge graph to unify the representation of data, software, hardware, and workflow. The developed approach is applied to a few selected case studies.

To realise a self-driving chemical laboratory, the design-make-test-analyse cycle is first reformulated as the process of propagating information through a dynamic knowledge graph by means of a chain of actions. Our approach utilises ontologies to capture the data and material flows involved in the experimentation, and employs autonomous agents as executable knowledge components to carry out both computational and physical tasks. The iterative workflow is automatically managed by a derived information framework with data provenance semantically preserved following the FAIR principles – Findable, Accessible, Interoperable and Reusable. The derived information framework is also applied to an automated flood impact assessment in the smart cities domain to demonstrate its generalisability. On the computational front, the dynamic knowledge graph approach is applied to automate the calibration of a kinetic reaction mechanism, which demonstrated a reduction of calibration time from months when done manually to days while an increase in accuracy measured as a 79% decrease in objective function value. In the wet lab, we demonstrate the practical application by linking two robots in Cambridge and Singapore to achieve a collaborative closed-loop optimisation for an aldol condensation reaction in real time. The two robots effectively produced a Pareto front for the cost-yield optimisation problem over the course of three days of operation. The dynamic knowledge graph approach is also applied to optimise two Suzuki coupling reactions for efficient synthesis of challenging molecules, obtaining 82 mg of the final product from a 40 mL scale-up that would be otherwise difficult to access without extensive process redesign and manual synthesis efforts.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Credit: rawpixel.com on Freepik

> *"Certainly, networked society will act as a whole, as an organism. There will be parameters of measurement of restlessness and stability analogous to hormone levels or body temperature of the human organism. But the analogy is of limited use, as we cannot tell what will come of the great connected system. Perhaps the possibility of global intuition will solve or find new problems posed by members."*

– Tim Berners-Lee, World-Wide Computer (1997), Communications of the ACM, 57–58

1.1 Motivation

The automation of laboratory involves linking the abstract concepts of chemical processes and the hardware responsible for the execution [382, 143]. It can be achieved by creating a fully connected virtual representation of the physical equipment and their status, *i.e.*, a "digital twin" of the laboratory that bridges the gap between the virtual and the real world. By doing so, it enables the orchestration of physical and computational experimentation in cyberspace [352], facilitating the automation of chemical discovery and scale-up. Therefore, it shortens the time span from making a new chemical in the research environment to the delivery of its mass production to the end-users. This presents the opportunity to deliver a significant level of decarbonisation with reduced labour and energy consumption, making the digitalisation of chemical manufacturing one of the critical technology paths towards a more sustainable society [170, 41].

Numerous advancements in automation for chemistry emerged during the late 1960s. These included significant milestones such as the first complete automation hardware for molecular synthesis [232, 233], the first expert system for generating scientific hypotheses called the Dendral project [205, 88], and the first retrosynthesis planning expert system, *i.e.*, the Logic and Heuristics Applied to Synthetic Analysis (LHASA) system [62, 63]. In the mid-to-late 2000s, the idea of a "robot scientist" also originated which integrates Artificial Intelligence (AI) and laboratory robotics for autonomous discovery of scientific knowledge in biology [189, 190]. Since then, considerable advances have been made to expand the potentialities of such a tool, covering the field of chemical reactions [60, 61], drug discovery [315], and material discovery for clean energy [350, 342]. For a detailed historical excursus, the readers refer to Dimitrov et al. [83].

Recently, laboratory automation has been reinterpreted as Self-Driving Laboratories (SDLs) with an emphasis on the closed-loop operation [146, 1]. In 2018, the participants of the Clean Energy Materials Innovation Challenge workshop recommended the development of Materials Acceleration Platforms (MAPs) [8], a platform-based approach, as the paradigm to accelerate the material discovery process. The concept of MAP was further expanded by Flores-Leonar et al. [108] to align with the realisation of SDLs. In line with the three key capabilities that were identified for a robo-chemist [278], *i.e.*, access to database of chemical reaction knowledge, synthetic steps planning, and automated execution of proposed action sequence, Flores-Leonar et al. [108] envisaged integration of Machine Learning (ML) algorithms and robotics platforms, with further interfacing between humans and robots, is the way towards autonomous experimentation. The current practices of development towards SDLs are seen following this trend. Researchers adopt automation of chemical experiments and advances in ML to enable functional material discovery [221, 209], the

discovery of chemical reactions [228], synthesis planning [347, 59], and optimisation of process conditions [106, 17, 35]. Despite the great success demonstrated by the community, the effort required to incorporate new equipment into an existing platform can be expensive. Tailored Extraction-Transformation-Loading (ETL) tools and the specific data exchange scheme for establishing effective communication are to be developed for each piece of equipment added. Therefore, these platforms normally face difficulties in scalability and interoperability due to heterogeneous data formats as an obstacle to holistic integration.

This problem is further exacerbated when it comes to the growing consensus within the scientific community that a paradigm shift towards a globally connected research network is necessary [342, 77, 207, 223]. This shift requires integrating distributed facilities from within one organisation [391, 222], and glueing different research groups to contribute their expertise towards solving emerging problems [326, 346]. Such integration also holds great potential in connecting earth- and space-based SDLs, so as to support human exploration in deep space [311]. As a prerequisite condition towards achieving this vision, the absence of standardised data representation and exchange protocols is seen as one of the critical challenges faced by the community [61].

A way forward may be offered by semantic web technologies [21], which present a vision of a fully linked web of data, demonstrating interoperability across scales and domains. It uses ontologies to describe the concepts and relationships within a given domain for communal understanding. In this thesis, we refer to ontologies developed to describe knowledge in the chemistry domain, and more importantly, those implemented in a way that is compatible with the semantic web standards [374], as chemical ontologies. One prominent example is ChEBI [148, 149]. An ontology normally consists of two components: a Terminological Box (TBox) and an Assertional Box (ABox) [374]. TBox refers to the description at a conceptual level, while ABox stores the data that is a realisation of the concepts defined by the TBox. Both levels can be accessed via Internationalised Resource Identifiers (IRIs), essentially generalised Uniform Resource Identifiers (URIs), for unambiguous identification. In the context of SDLs, this opens up the possibility of developing a fully linked data representation for the chemical processes and equipment status as a universal framework to facilitate concrete data exchange within and between laboratories [112].

Besides the interoperable data representation, an effective way to communicate and share data must be addressed to achieve laboratory automation. In this regard, collective intelligent agents have been used to automate the tasks involved in crystal-structure phase mapping [129], material discovery [241], and reaction optimisation [43]. Considering the historical discussions of integrating the two technologies [154], we hypothesise that a

dynamic Knowledge Graph (dKG), *i.e.*, an ontological representation of a laboratory that is constantly updated by AI agents, could expedite the implementation of networked SDLs.

Prior to delving into this technology, it is important for us to comprehend the challenges we aim to address through its implementation. Achieving the vision of a globally connected network of SDLs is not an easy task and entails four major challenges. The first challenge is efficiently orchestrating heterogeneous resources [32, 284], which includes hardware from different vendors and diverse computing environments. The second challenge is sharing data across organisations [61, 146], which requires standardising language in which the research is communicated [382]. During this process, the source and metadata of the research need to be tracked to facilitate reproducibility, which leads to the third challenge of developing workflows that produce data adhering to FAIR principles – Findable, Accessible, Interoperable and Reusable [383, 174]. Finally, establishing an open infrastructure is crucial to enable the dynamic inclusion of diverse perspectives as the network expands, fostering future-proof science [207, 275].

Orchestration of heterogeneous resources

This challenge lies in the management and automation of scientific workflows encompassing various SDLs [223]. Different groups normally build SDLs with software and hardware from diverse vendors. These resources process at varying levels of fidelity and have the capacity to conduct experiments at different time and length scales. Despite ongoing efforts in modularisation, achieving seamless interoperability among the components remains largely unresolved. Furthermore, the adaptability of the workflow is crucial, as it must accommodate alterations in the scope and objectives of experiments [191], as well as dynamically incorporate new SDLs into the network as the experimental campaign progresses [223].

For contemporary SDLs, *i.e.*, those following a platform-based approach, this poses a heavy workload on the central software to coordinate the information flow. Different views on the solution to this challenge co-exist. Standardisation in Lab Automation (SiLA) [313] and Laboratory and Analytical Device Standard (LADS) [33] are initiatives to provide standardised interfaces for proprietary hardware, adopting a mindset of peer-to-peer information exchange. These approaches partially alleviate the burden of the central coordinator by enforcing that each peer in the network comprehends the communication protocols used by all the others. In contrast, Roberts et al. [300, 301] advocates a vision where information is stored in a centralised location that is accessible to all stakeholders within a laboratory environment. By allowing all entities to share the same worldview, this perspective flattens the structural design. The communication between peers only acts as a pointer to the correct resources. This unifies the peer interaction but puts strain on network infrastructure as all

access requests are now centralised. Further explorations are required in both directions to reveal a future of "plug-and-play" SDLs.

Data generation, integration, and utilisation

This challenge lies in the data management practice across different scales of collaboration [275]. Going towards a fully digitalised network of SDLs, the ability to automatically capture all generated data within an experiment (even a "bad" reaction), integrate it with data from other groups, and share it with the broader community is crucial for navigating the chemical space. To this end, a consensual description of the experiment is required. Ideally, it should also enable retrospective integration with literature data. Despite various formats intending to standardise the reaction representation, they are currently in their early stages of development, resulting in a low level of practical adoption in the community.

Data generation is not an end goal in itself, it is more important to allow the community to discover new knowledge and form new hypotheses from the collected data. In addition to the absence of experimental representation, there is also a lack of a communal framework to facilitate the exchange of data. Although Electronic Laboratory Notebooks (ELNs) and relational databases are two solutions utilised in the industry that come close to fulfilling this purpose, their disconnection with SDLs is evident from both directions. Firstly, these ELNs and databases are designed as collections of recordings that are generated after the completion of experiments. Secondly, the implementation of SDLs often relies on *ad hoc* data representations. Both shortcomings necessitate researchers to invest significant effort in organising and preparing data that demands expertise in data engineering, a domain that is typically overlooked in the education and training of scientists.

Consequently, there is a pressing need for a technology that can establish a connection between SDLs and data repositories, allowing data to be digitally generated and recorded directly by machines. Such a technology would significantly enhance data accessibility and promote interoperability within the scientific community. By enabling data to be "born digital" [231], right from the moment it is captured, this advancement would facilitate seamless integration and utilisation of scientific data across various SDLs.

FAIR workflows to facilitate reproducibility

This challenge lies in contextualising data in scientific workflows. It expands on the scientific data (measurements/results) and emphasises the entire workflow, including capturing metadata of an experimental process, documenting workflow procedures, and recording environmental measurements. While SDLs facilitate real-time data sharing, preserving the

provenance of the data is equally crucial. It is imperative to develop FAIR workflows that can be traced to ensure reproducibility [125], which is essential for accelerating scientific discovery. Similar to making data FAIR [12, 78], achieving FAIR workflows should not be a one-time effort but rather implemented throughout the entire research process. An automated solution is needed to capture provenance information automatically as the SDLs themselves conduct the experiments. This encompasses documenting exogenous ambient conditions and the equipment used in the experimental setup. On the computational front, how certain values were calculated from which dataset is also important to version-control the resulting models [293]. By doing so, it is possible to reduce human error stemming from inconsistent and incomplete documentation practices among individual researchers. Ultimately, addressing this challenge would contribute to enhancing research integrity and facilitating scientific progress. For instance, we should be able to answer these types of questions:

- What historical data were utilised in this Design of Experiments (DoE) study?
- What are the instrumentation parameters configured for this piece of hardware at any given time?
- What sub-tasks should be executed after the current sub-task to complete the predefined workflow?

Open infrastructure for open science

To address the aforementioned challenges, an open infrastructure that seamlessly integrates data, software, hardware, and workflows across SDLs in a unified manner is needed. Providing the community with essential tools for structuring data at its generation is imperative. Additionally, new software solutions should be evaluated on public datasets to benchmark their performance. Standardisation in hardware data acquisition is necessary to enable easy incorporation of new equipment. Moreover, this infrastructure should facilitate the "plug-and-play" of new SDLs seeking to join the network. Ideally, an off-the-shelf solution compatible with any platform should be available to reduce the barriers to entry. Therefore, interoperability is key towards establishing such an open infrastructure for open science.

Developing such a usable and reusable technological solution is an iterative process that necessitates engagement from academia, industrial vendors, government entities, and the general public. It is envisioned to be a community endeavour throughout the entire life-cycle of the development. Trial-and-error will be inevitable in the coming decade. Therefore, such an initiative should be an open-source project, with all resources available through public

repositories such as GitHub and GitLab, welcoming contributions from the community. From an educational perspective, such democratisation is vital to avoid students being reliant on proprietary products during their education, which may impede their ability to leverage their expertise after graduation.

1.2 Aim and scope

The main aim of this thesis is to investigate the suitability of a dynamic knowledge graph in addressing the challenges encountered by contemporary SDLs. The thesis formulates a dKG approach along with its underlying rationale and applies it to selected case studies to showcase its effectiveness in all four aforementioned challenges.

1.3 Novel contribution of the thesis

This dissertation presents the following novel contributions:

- A comprehensive review was conducted for the data format and transfer protocols adopted in the contemporary implementation of SDLs, as well as the community efforts in standardisation and digitalisation of chemistry knowledge. This analysis revealed the need to transition towards a dKG approach for enhanced interoperability.
- A KG-native solution was developed to manage the derived information in a dKG. The workflow of computational and physical processes was abstracted as information flowing in KGs. The framework is generic and it is applicable to various use cases, including smart cities and SDLs.
- The utilisation of High-Performance Computing (HPC) in the automated calibration framework for kinetic mechanisms demonstrated significant improvements in both time efficiency and accuracy when compared to manual model development. This application presented a step towards automated model management throughout its life cycle.
- A proof-of-concept was conducted to demonstrate the closed-loop optimisation of an aldol condensation reaction in a decentralised manner. The dKG demonstrated a real-time collaboration connecting two laboratories that are more than 10,000 km apart. This proof-of-concept shed light on an infrastructure to support the generation and sharing of FAIR data at a community scale.

- Two Suzuki coupling reactions were optimised using the dKG approach and achieved an efficient synthesis of challenging molecules. The optimisation involved both continuous and categorical variables for catalyst selection. This case study demonstrated the broad capability and effectiveness of the dKG approach.

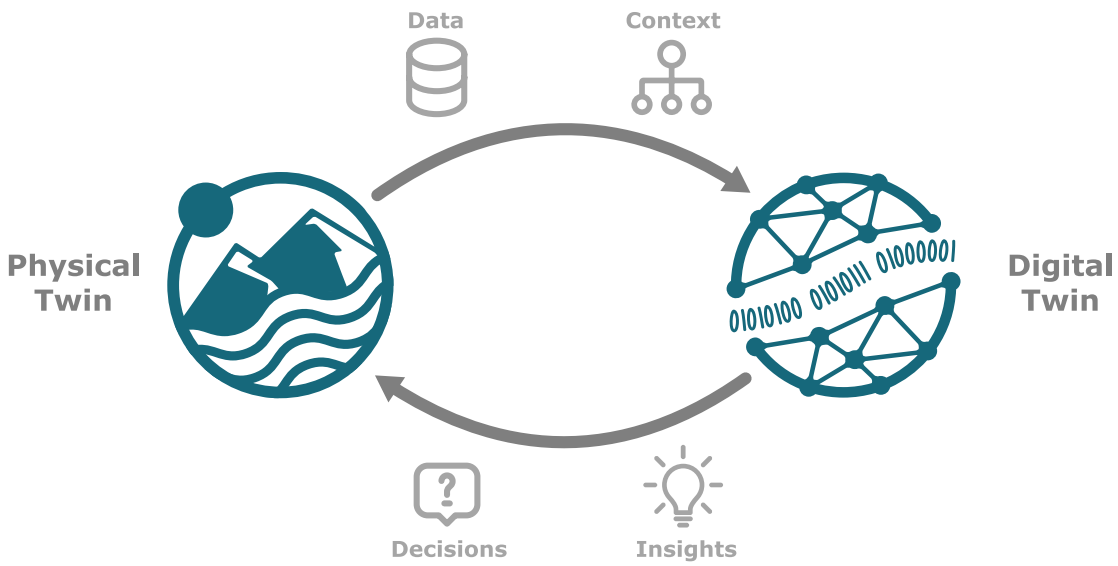
1.4 Structure

This document is organised into seven chapters. After some necessary background introduction, Chapter 3 identifies the necessity of a transition from the current implementation of SDLs to the adoption of dKG technology. Chapter 4 then develops a KG-native solution to address the challenges. Chapters 5 and 6 apply the dKG approach in computational and experimental studies respectively. The final chapter summarises the work and proposes further studies.

- Chapter 1 – *Introduction*
- Chapter 2 – *Background* provides some background on SDLs and key concepts to semantic web and agents, which leads to the introduction of the World Avatar project.
- Chapter 3 – *Review of community efforts in digitalisation of chemical research* conducts an extensive analysis of the current community landscape in data management of SDLs from both the practical implementations and the initiatives in data formats and exchange protocols.
- Chapter 4 – *Automated provenance annotation and update* develops the fundamental infrastructure for abstracting both computational and physical experimentation tasks as an information derivation process by semantic agents, reformulating workflow execution as agents' actions in a dKG.
- Chapter 5 – *Automated mechanism evaluation and calibration* presents the utility of dKG in the computational model development aspect of an SDL.
- Chapter 6 – *Automated experiment execution and optimisation* demonstrates the utility of dKG in the wet-lab experiments of one SDL and across geographically apart SDLs.
- Chapter 7 – *Conclusions* summarises the thesis and considers the direction of future works.

Chapter 2

Background



Credit: theworldavatar.io

> *"... it seems probable that once the machine thinking method had started, it would not take long to outstrip our feeble powers. There would be no question of the machines dying, and they would be able to converse with each other to sharpen their wits. At some stage therefore we should have to expect the machines to take control..."*

– Alan Turing, Intelligent Machinery, A Heretical Theory (1996), Philosophia Mathematica, 256–260

In this chapter, we begin with the background of SDLs which is primarily based on the well-recognised literature [8, 146, 108, 327]. We then introduce the historical background of semantic web technologies and software agents, much of which is found in computer science texts. Finally, we introduce the World Avatar project. This outlines the vision of dKG technology which combines KGs and software agents. It also situates this thesis in a bigger picture of the research initiative.

2.1 Self-driving laboratories

Combining AI algorithms with autonomous experimentation, SDLs hold the promise of expediting the process of scientific discovery [327]. It is also envisaged by von Lilienfeld and co-workers that SDLs will emerge as the fifth pillar in the domain of hard science [167]. This thesis discerns the conceptualisation of SDLs evolved in approximately three stages. Initially, the concept of MAPs specified six integrated goals for the next-generation materials discovery [8]. Shortly after, the concept of SDLs is formalised with an emphasis on closed-loop operations [146]. This further leads to the formalisation of the five core components of MAPs, offering guidelines for its adoption across the scientific community [108]. While a succinct overview of each stage is provided here, a more comprehensive thought process can be found in the original works referenced.

2.1.1 Conceptualisation of MAPs

MAPs represent an ambitious initiative geared towards shortening the materials development cycle, condensing it from a decade or two to a mere one or two years [8]. This concept was introduced as part of the Clean Energy Materials Innovation Challenge workshop organised by Mission Innovation members in 2017. During the workshop, participants drew inspiration from successful implementation in the biological and medical sciences, particularly focusing on components of robotic platforms that exhibited greater precision and research efficiency. This cross-disciplinary experience shed light on the limitations of the current approaches in materials science, which lack the full-scale deployment of autonomous discovery tools. This analysis culminated in the identification of key gaps and the proposal of six interconnected priority research areas, denoted as the *Six Grand Goals* (see Fig. 2.1), aimed at bridging these gaps and advancing materials research. The original wording is quoted below in bold with our commentary:

- **Closing the loop in autonomous discovery and development:** This goal emphasises the aspiration to fully integrate MAPs into laboratory settings, enabling them to inde-

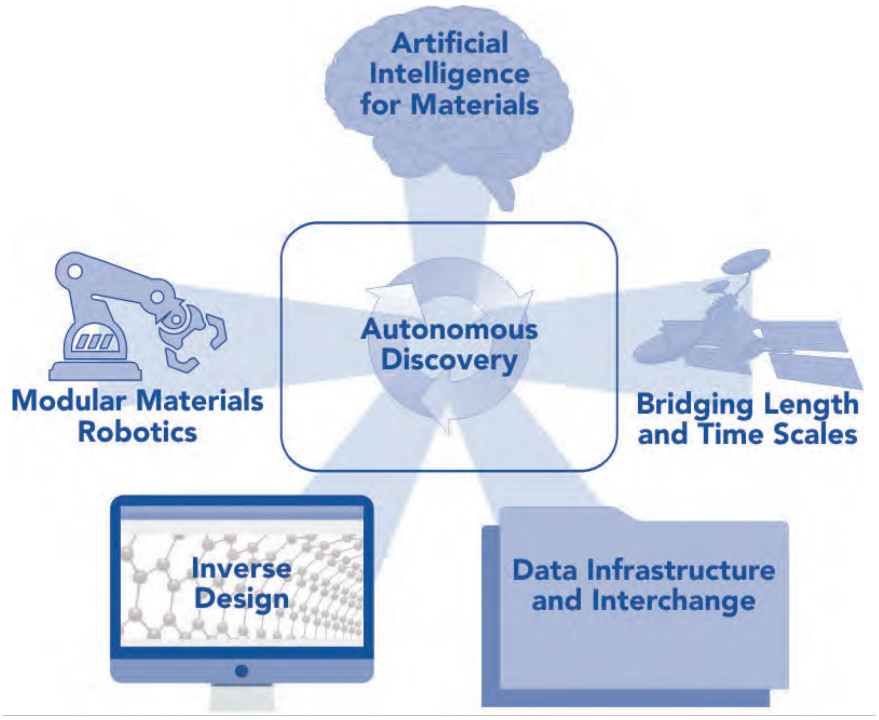


Fig. 2.1 Six grand goals of the MAPs. Adopted from Aspuru-Guzik and Persson [8] © under a CC BY 4.0 licence.

pendently manage experiment design, execution, and result interpretation throughout both the materials discovery phase and subsequent scale-up processes.

- **Artificial intelligence for materials:** This objective refers to the necessity of empowering machine systems with human-like cognitive capabilities that are required in various steps of autonomous discovery. The international community should work towards encoding human intuition into machines and advancing their capabilities to supersede human intelligence.

- **Modular materials robotics:** Recognising the dynamic nature of scientific discovery, it is crucial to develop adaptable robotic components resembling "Lego-like" bricks for easy assembly within MAPs. This modularity facilitates efficient exploration beyond known materials.

- **Inverse design:** Addressing global challenges such as climate goals requires a systematic approach where the community needs to backcast the optimal material properties and the subsequent identification of appropriate candidates possessing these attributes. The inverse design approach will be more efficient in this task compared to the conventional Edisonian approach.

- **Bridging length and time scales:** The development of new materials encompasses events occurring at various time scales. This requires a unified architecture that connects data and ideas across these scales to enable efficient collaboration among researchers.
- **Data infrastructure and interchange:** Effective communication and collaboration with the international community are vital for tackling global challenges like the development of clean energy materials. To achieve this, a robust data infrastructure is imperative to facilitate seamless data exchange and cooperation.

The workshop concluded that developing clean energy materials to achieve the net-zero target is a global challenge that necessitates a worldwide response. It emphasises the need for increased financial support and active involvement from students, policymakers, international leaders, investors, and the general public. At this stage, while the ultimate goal is clear, the exact technology solution for achieving it requires further thought.

2.1.2 SDLs for closed-loop operations

The idea of SDLs was proposed by Häse et al. [146] in 2019. The key factor is a closed-loop approach where the AI models (planning algorithms) are integrated with the automated robotics for experimentation (see Fig. 2.2).

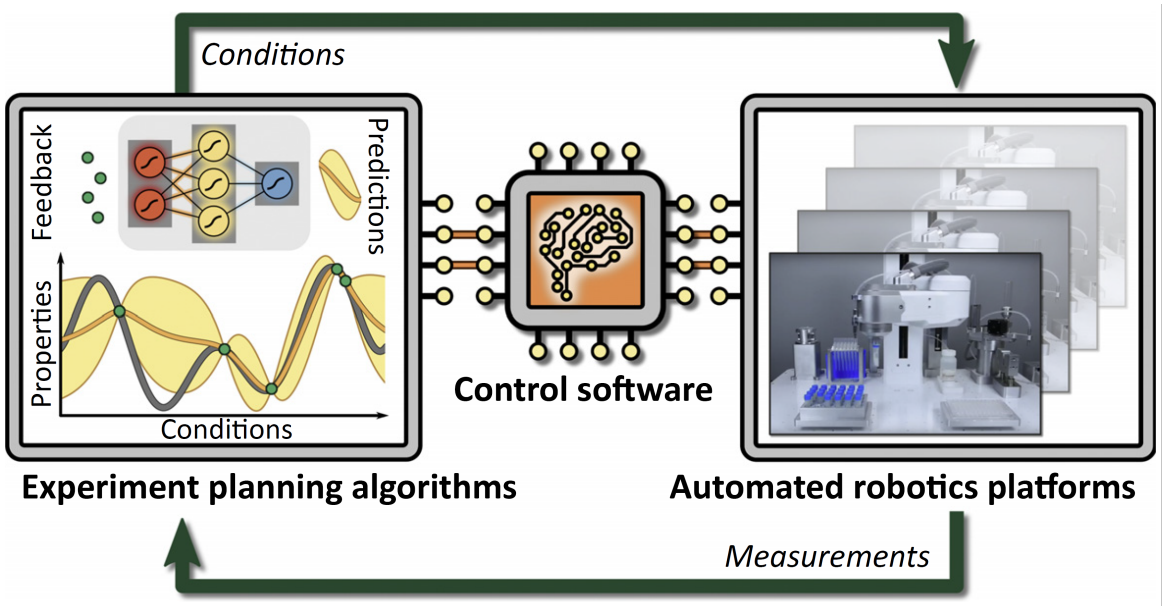


Fig. 2.2 The closed-loop approach for automated experimentation [146]. Used with permission from Elsevier ©.

Figure 2.3 outlines the modules essential for realising an SDL from a functional perspective [146]. These modules fall into two categories based on their role in the experimentation process. The first category includes experiment planning and the orchestration of heterogeneous experimentation platforms that are integral to the closed-loop approach and hence directly influence the discovery rate. The second category focuses on improving human-machine interaction and long-term maintainability. It consists of the management, processing, and communication of experimental findings. In practice, an SDL may consist of only a subset of the complete range of modules. Nevertheless, it is crucial for the community to standardise the development and prototyping of individual modules to facilitate the roll-out of SDLs.

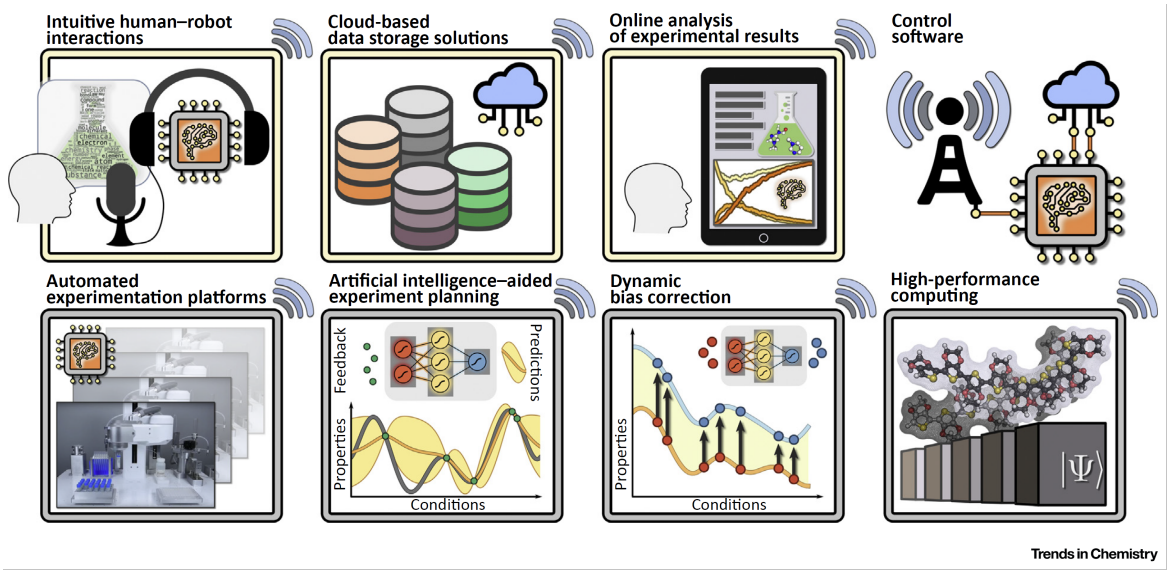


Fig. 2.3 The modules involved in SDLs [146]. The modules highlighted in grey are directly involved in the experimentation process whereas those highlighted in yellow improve the useability and maintainability of the SDL. Used with permission from Elsevier ©.

When drawing upon ideas from the drug discovery workflow, the closed-loop operations can be subdivided into four steps, namely: Design, Make, Test, and Analyse (DMTA). The same procedural framework is adaptable to materials discovery, as illustrated in Fig. 2.4. Typically, this process commences with the specification of the desired material property as the objective. An inverse design algorithm then generates a suitable candidate material that is anticipated to possess the desired properties. The synthesis phase is executed by the robotic platform, which in turn initiates the characterisation process. The resulting measurements are subjected to analysis and integrated with previous data to inform the design of the subsequent iteration. As depicted in the illustration, the focal point of this iterative workflow is the data infrastructure for managing information flow throughout the loop.

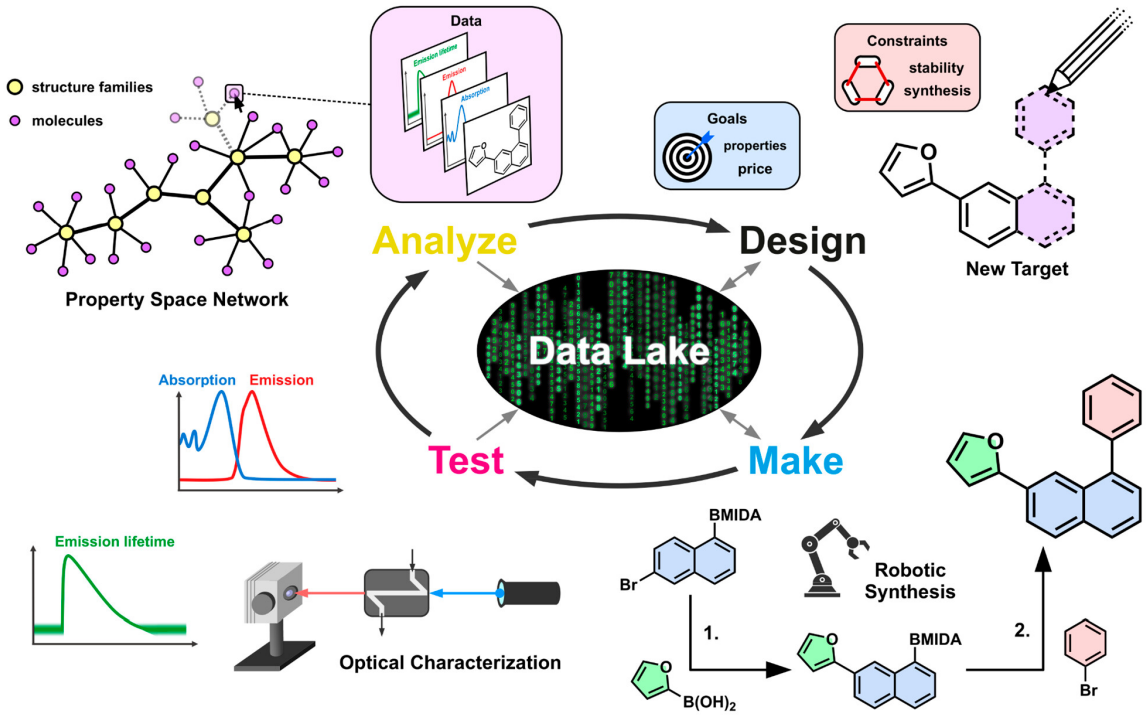


Fig. 2.4 Interpretation of closed-loop workflow in SDLs as the design–make–test–analyse cycle. Adopted from Seifrid et al. [327] © under a CC BY NC ND 4.0 licence.

2.1.3 Formalisation of functional components

The abstraction of MAPs is further matured when considering the components necessary to implement an SDL. Figure 2.5 illustrates the five key elements of MAPs function within a closed-loop approach as outlined by Flores-Leonar et al. [108] in 2020: human intuition, AI models, orchestrator, robotic platforms, and databases. The AI models suggest the next experimental conditions for execution by the robotic platforms. All data generated during these experiments are subsequently stored in databases. The orchestration of this entire process is overseen by the orchestrator, which facilitates the flow of information throughout the workflow.

At this point, SDLs and MAPs become somewhat interchangeable. This type of abstraction has also been adopted by numerous projects, including noteworthy examples beyond the original author's group, such as BioMAP [351], Polybot [373], and AlphaFlow [371]. Many researchers have also engaged in discussions about future directions based on their own experience. These discussions cover a wide range of topics, including dispensing a small amount of solid [327], the integration of diverse instrumentation sets required by different workflows [274], the potential risks associated with employing AI in laboratory settings [250], and many other considerations [223].

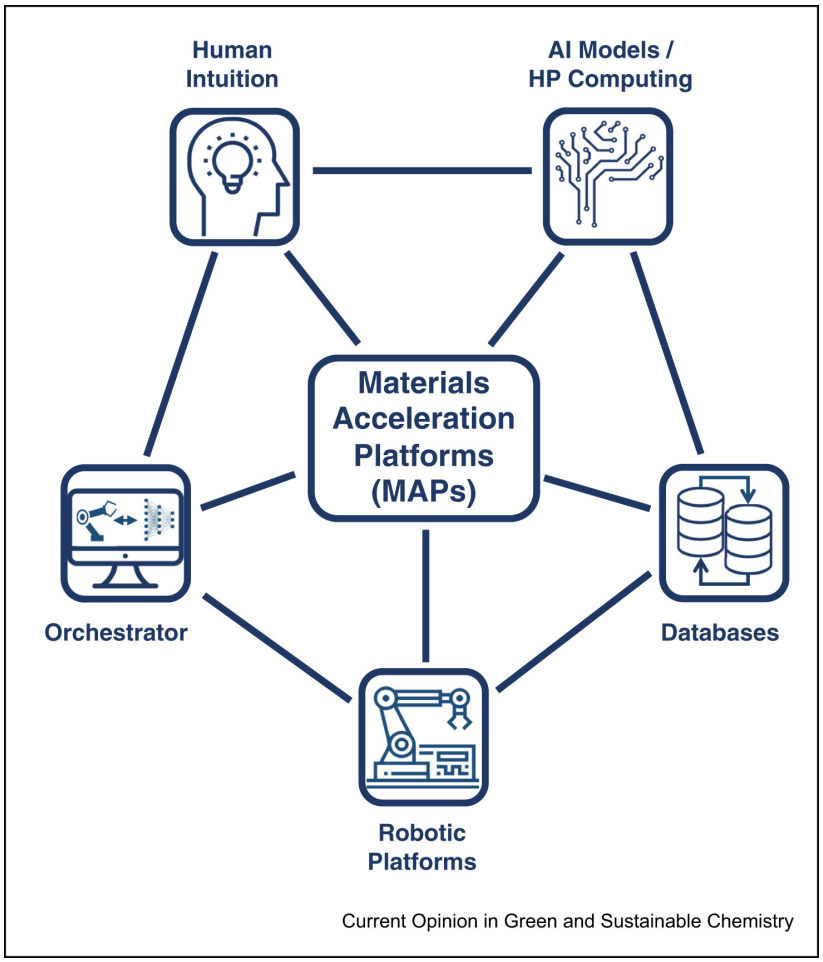


Fig. 2.5 Five functional components that comprise the MAPs [108]. Used with permission from Elsevier ©.

In resonance with the central role occupied by the data lake in Fig. 2.4, this thesis focuses primarily on the interoperability of chemical experiments, with a specific emphasis on enabling closed-loop operations that could potentially be distributed. The fundamental challenge in achieving this goal lies in the efficient management of data and information flow between distinct modules. The implementation of such data infrastructure of the selected contemporary SDLs will be thoroughly analysed in Chapter 3.

2.2 Semantic web and agents

Since the landmark publication by Berners-Lee et al. [21], the semantic web community has envisioned the next generation of the web in both a human- and machine-readable format for better data sharing among mankind and faster data processing using computers. Through ups

and downs, the focus of the semantic web community has pivoted from ontologies to linked data, and further to knowledge graphs, which are gaining attention again in recent years [157]. Below, we provide a brief overview of each phase [157] and explore how software agents can be integrated with the semantic web in accordance with its initial vision [21]. Readers interested in anecdotal insights may refer to the talk of Hendler [155].

2.2.1 Ontologies

In the early stage of developments in semantic web technologies (from the late 1990s to mid-2000s), *ontology* is the central notion adopted by the community. As a specialised form of explicit knowledge representation [135], the primary objective of an ontology is to provide a well-defined and widely accepted knowledge representation within a specific domain of interest. It defines relevant concepts, often termed as *classes*, the relationships between these concepts, which are known as *properties*, as well as rules and restrictions that describe these relations. Properties link concepts from *domain* to *range*. Specifically, *object properties* connect two classes and *data properties* link a single class to data *literals*. An ontology is normally comprised of two primary components: a TBox that provides descriptions at a conceptual level and an ABox as the repository for the actual data that materialises the concepts established by the TBox [374]. This facilitates the sharing of knowledge and data by experts and practitioners within that specific domain.

Web Ontology Language (OWL) is a widely accepted form for representing ontologies. It was introduced as a World Wide Web Consortium (W3C) standard in 2004 and later updated as OWL 2 [375] in 2012. To keep the writing concise, the term "OWL" will be used to represent its latest version, *i.e.*, OWL 2, throughout the remainder of the thesis. This principle also applies to other standards introduced in this section. OWL is based on *description logic*, a sublanguage of first-order predicate logic, making logical deductive reasoning over the language decidable. In competition with description logic, a rule-based language known as Rule Interchange Format (RIF) [186] was also standardised by W3C. However, it failed to gain widespread traction compared to the dominance of OWL within the community.

Resource Description Framework (RDF) is another critical standard in the realm of the semantic web. It was also adopted as a W3C standard in 2004 and subsequently updated to RDF 1.1 [317] in 2014. RDF is a syntax for expressing information on the web as directed, labelled and typed graphs. RDF/XML is the default serialisation format for OWL, and its support is mandatory for all software conformant to OWL. RDF/XML was initially designed to be processed by XML (stands for eXtensible Markup Language) tools, however, it never became popular within the XML community due to the potential complexity and the difficulty of processing it [87, p. 27]. As an example, consider three facts below:

- The paper with doi:10.1145/253671.253704 has the creator Tim Berners-Lee [18].
- The paper with doi:10.1145/253671.253704 has the title “World-Wide Computer”.
- Tim Berners-Lee’s title is “Director”.

They can be represented in RDF/XML format:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:v="http://www.w3.org/2006/vcard/"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

    <rdf:Description rdf:about="https://doi.org/10.1145/253671.253704">
        <dc:creator rdf:resource="http://www.w3.org/People/Berners-Lee/card#i"/>
        <dc:title>World-Wide Computer</dc:title>
    </rdf:Description>

    <rdf:Description rdf:about="http://www.w3.org/People/Berners-Lee/card#i">
        <v:title>Director</v:title>
    </rdf:Description>

</rdf:RDF>
```

Alternatively, the elements in the above representation can be nested, also known as *striping*, and will appear as follows:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:v="http://www.w3.org/2006/vcard/"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

    <rdf:Description rdf:about="https://doi.org/10.1145/253671.253704" dc:title="
        World-Wide Computer">
        <dc:creator>
            <rdf:Description rdf:about="http://www.w3.org/People/Berners-Lee/card#
                i">
                <v:title>Director</v:title>
            </rdf:Description>
        </dc:creator>
    </rdf:Description>

</rdf:RDF>
```

To ensure a consistent understanding of the information in this small "dataset", eXtensible Stylesheet Language Transformations (XSLT) and other XML processing tools would have to check various locations, *i.e.*, elements and attributes, which is not scalable in practical applications. Consequently, various serialisation syntaxes were developed to address different needs. For instance, OWL/XML [249] is designed for simpler processing using XML tools, Manchester syntax [166] is developed for better handling of description logic, and Turtle [16] is a compact yet human-readable format for creating and parsing RDF triples. These options allow developers to choose the most suitable representation based on their specific requirements and preferences. Turtle format has been chosen for the remainder of the thesis for brevity. This choice is evident when we compare the following Turtle expressions for the three facts:

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix v: <http://www.w3.org/2006/vcard/> .

<https://doi.org/10.1145/253671.253704>
    dc:creator <http://www.w3.org/People/Berners-Lee/card#i> ;
    dc:title "World-Wide Computer" .

<http://www.w3.org/People/Berners-Lee/card#i>
    v:title "Director" .
```

The data expressed in any RDF serialisations can be queried via SPARQL (pronounced 'sparkle'), a recursive acronym for SPARQL Protocol and RDF Query Language. Its first version became an official W3C specification in 2008 and an update was formalised as SPARQL 1.1 W3C Recommendations in 2013 [359]. A SPARQL query that corresponds to the question *who is the author of the paper titled "World-Wide Computer"* is exemplified below:

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>

SELECT ?author
WHERE {
  ?paper dc:title "World-Wide Computer".
  ?paper dc:creator ?author.
}
```

During this time, numerous tools emerged to ease the handling of ontologies. Prominent open-source examples are Protégé [254], a graphical tool introduced in 1999; OWL API [165],

a Java application programming interface (API) initially launched as part of the WonderWeb Project between 2002 to 2004; and RDFLib [198], a Python library first released in 2002, to name a few.

A noteworthy trend during this period was the emphasis on creating deep and complex ontologies, particularly within the fields of biology and healthcare. However, despite initial achievements, *e.g.*, the formal OWL release of Gene Ontology [358] and SNOMED CT [318], the community encountered significant upfront costs when constructing large-scale ontologies, and these ontologies often proved challenging to be reused by others. As a result, a change in sentiment emerged within the community during the mid-2000s, leading to a shift away from ontologies.

2.2.2 Linked data

The idea of *linked data* was proposed by Berners-Lee [19] in 2006 and remained the main driver of the community until the early 2010s. It is not a new standard, but rather a set of "rules" which consists of four principles:

- Use URIs as names for things.
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL).[1]
- Include links to other URIs so that they can discover more things.

When the linked data is released under an open licence, the idea is expanded to *linked open data*. As opposed to the focus on the depth of the TBox in the previous phase, the emphasis of these specifications is on the volume of the data, *i.e.*, ABox, that is free and publicly accessible. Ontologies still play a role, but most of the linked data adopts a shallow and non-expressive ontology. Some well-known examples during this era are DBpedia [206], Schema.org [137], Wikidata [276], and Linked Open Data Cloud [173], to name a few.

It was also around this time that the US and the UK governments started to publish more data on their official website. Berners-Lee [20] suggested a five-star deployment scheme and promoted the governments to publish their data with semantic annotations. This initiative to some extent affected the governments' decisions, exemplified by the appointment of

[1]Note that the asterisk in "RDF*" means that it refers to both RDF and RDF Schema (RDFS) as standards. This is to be distinguished with RDF-star [389], which is a work in progress to support statements about statements, *i.e.*, one can model provenance and other structured metadata for an edge in a graph.

Professor James Hendler as the "Internet Web Expert" for Data.gov [87, p. 42]. At the time of writing, around 5% of the dataset[2] published by the US government is available in RDF format [365]. The UK government also provides SPARQL endpoint for some of its data, for instance, Office for National Statistics [263].

In the early 2010s, the initial enthusiasm around linked data gradually evolved into a more pragmatic view [157]. Publishing data in the form of linked data is undeniably positive by making the data accessible in a machine-readable format. However, the community also acknowledged the challenges tied to the curation and integration of linked data sourced from external providers. This promoted a further quest for more efficient approaches to data sharing and reusing.

2.2.3 Knowledge graphs

In 2012, Google launched its *knowledge graph* to improve its search efficiency [334]. A visible product of this "new technology" is the so-called Google Knowledge Panel that appears next to the search bar when searching for entities that are in the KG. Figure 2.6 exemplifies such a panel for a search on *Marvin Minsky*. Users can click through the hyperlinks to access related panels. The Google KG is not available for public download but offers a read-only API [132] for developers to access information programmatically. This graph adopts JavaScript Object Notation for Linked Data (JSON-LD) [340] as its serialisation format, another W3C standard that is compatible with RDF and can be embedded into HyperText Markup Language (HTML). Other technology companies in the field are also actively using KGs in their products [261], including Microsoft, Facebook (now Meta), eBay, and IBM.

From a technological standpoint, the term "knowledge graph" essentially represents a re-branding of established concepts within the semantic web field. Nonetheless, the focus within this phase has notably shifted. Specifically, the industry now overtakes the academic institutions in its leadership in this wave. This transition has led to two primary distinctions: Firstly, industry-scale KGs are typically proprietary, diverging from the previous open approach. Secondly, there is a shift in control. Major corporations now possess the data and consequently instantiate it following their own schema. This stands in contrast to the linked data era, where the emphasis was on establishing connections with external sources but less on the expressiveness of schema.

Other interpretations of KGs also exist. One notable example is the *labelled property graph*, originally developed by a team of Swedish engineers with the aim of facilitating fast

[2] 12,230 out of 236,506 datasets are available in RDF format as of 14 September 2023.

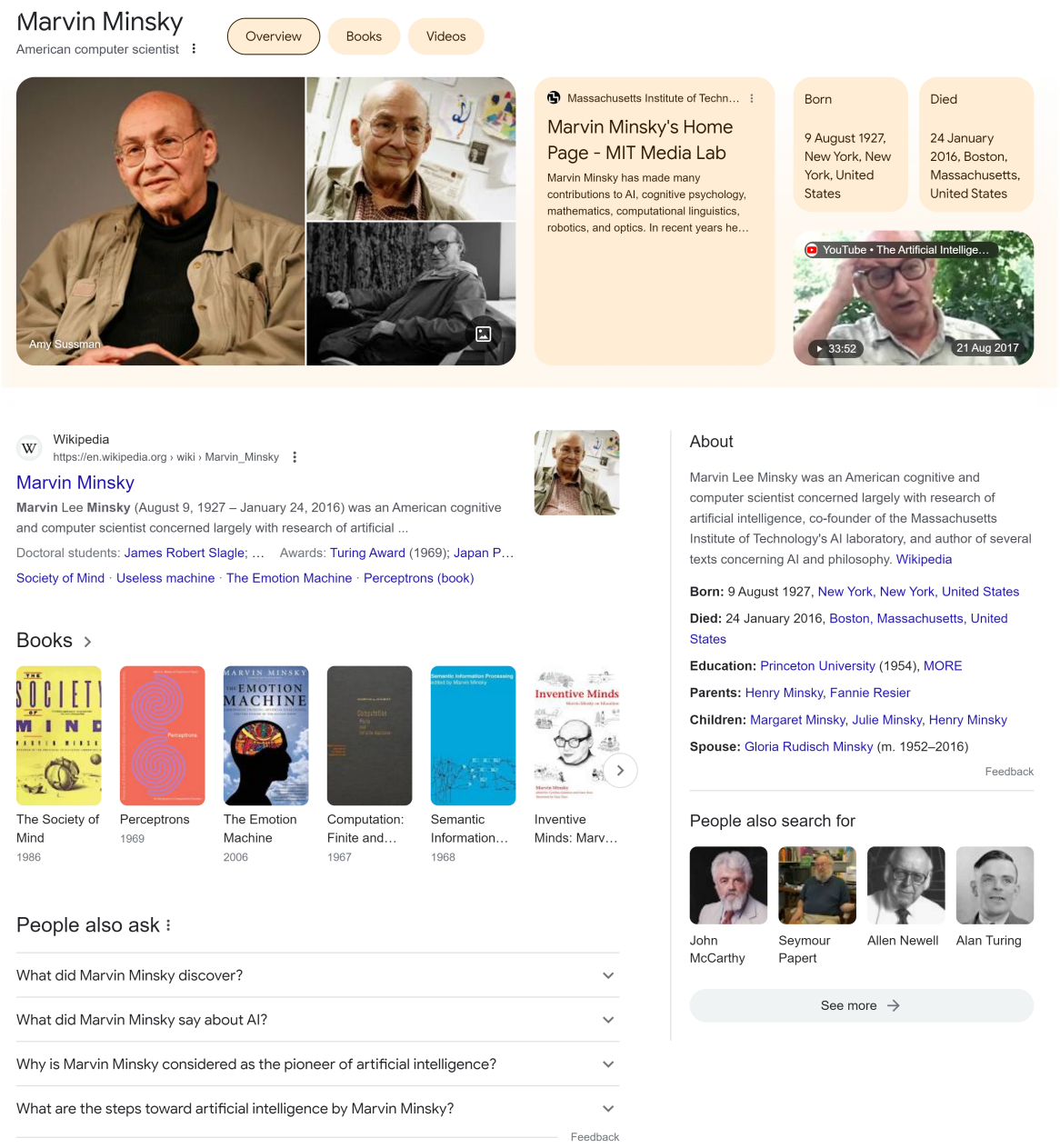


Fig. 2.6 Google knowledge panel results as shown after searching on https://www.google.co.uk/ for the term *Marvin Minsky*, 12 September 2023.

traversals across connected data [14]. This concept eventually led to the establishment of the company Neo4j and their graph database product. The labelled property graph offers an advantage over RDF in terms of simplicity and efficiency when it comes to adding edge qualifiers. In fact, directly adding qualifiers to an edge in RDF is still a work in progress [389]. However, its main focus on information retrieval for graph analytics leads to the utilisation of

local identifiers for nodes. This implementation limits its suitability for large-scale distributed applications, where the use of globally unique IRIs is a more desirable and practical solution.

Despite the shifted leadership and focus, the challenges encountered in semantic web technologies largely persist in the context of KGs. Ongoing research efforts continue to address these challenges, with significant collaboration between academia and industry evident at two key annual conferences in this field: the International Semantic Web Conference (ISWC) and the China Conference on Knowledge Graph and Semantic Computing (CCKS).

2.2.4 Autonomous agents

The term *autonomous agent* is defined as a piece of automated software programme capable of acting towards achieving its objectives [309]. Since the theory proposed by Marvin Minsky in his book, *The Society of Mind* [236], agents have strong connections with the field of AI. Minsky [236] suggests that intelligence does not originate from a singular, flawless principle but rather emerges from interactions among a group of individual agents, each endowed with specific functions. This perspective is supported by Russell and Norvig [309], who assert that AI arises from intelligent agents making rational decisions in certain situations. In this process, agents communicate, coordinate, and exchange information with each other in a standardised format. The ideas surrounding agents and their interactions have materialised in the field of *multi-agent systems* [387]. Within this domain, a central research question revolves around achieving effective coordination and collaboration among agents to solve problems. Various communication languages were introduced in this field, as seen in works like Knowledge Query and Manipulation Language (KQML) [103] and Agent Communications Language (ACL) [283]. However, these languages often had relatively rigid formats and limited semantic expressive capacities.

Following the introduction of ontological data representations in the semantic web technologies, a natural question is to ask whether the use of agents and ontologies can be combined to harness the strengths of both approaches. The challenge of how best to do this has been an open research question since the initial proposal of semantic web [21], where software agents are envisaged as personal assistants to humans. In theory [21, 154], ontologies can help agents with more flexible operations, whereas agents can help the ontologies for better data utilisation. In the sense of a multi-agent system, the semantic web is the "environment" of the agents, where ontologies are used to establish a common understanding of the topic of interest.

The Foundation for Intelligent Physical Agents [357] (FIPA) proposed a set of specifications focusing on communication and interoperability between agents. Specifically, the FIPA Ontology Service Specification detailed the concept of an ontology agent to facilitate message

interpretation among agents. However, it never made it to the standard stage. In the following years, Java Agent DEvelopment framework (JADE) [176], a Java-based software platform that simplifies the implementation of FIPA-compatible multi-agent systems, attempted to provide an ontology in its realisation of FIPA standards. However, they only provided the ontology as part of the Java code, without connecting to a knowledge base. Attempts to merge the two technologies have been seen in other domains, but not much in chemistry until very recently. An attempt to do this is described in the next section.

2.3 The World Avatar

The World Avatar[3] project was initiated by Markus Kraft and co-workers which aims to develop an all-encompassing *digital twin* [89, 3] that is capable of describing any aspect of the world. It uses a *dynamic knowledge graph* based on an ontological representation of entities and interoperable agents, as illustrated in Fig. 2.7. In this section, we outline the vision and status of the World Avatar project, followed by a discussion on its design philosophy–dynamic knowledge graph–from a more technical point of view.

Before diving into further details, we provide a glossary of terms that are heavily used in the World Avatar. We acknowledge that the terms may have different meanings in other contexts; we make no attempt at general definitions here.

Knowledge graph: a collection of data and software agents expressed as a directed graph controlled by ontologies, where the nodes and edges refer to concepts and relationships correspondingly. This has broader coverage than the KG as commonly used in semantic web studies as mentioned in the previous section, where only data are modelled as a directed graph [157]. This is also different from the KG built based on Reaxys by Segler and Waller [324] for reaction discovery problems, which expressed molecules as nodes and binary reactions as edges.

Digital twin: a virtual replica of real-world entities in the form of a KG. It is usually created for the real-time monitoring and controlling of real entities and, thus should be synchronous with its physical counterpart.

Dynamic knowledge graph: a KG that is constantly modified by agents with the latest status of the real world. It controls and influences the real world by updating the specifications of the digital twin and actuating that with agents.

[3] https://theworldavatar.io/

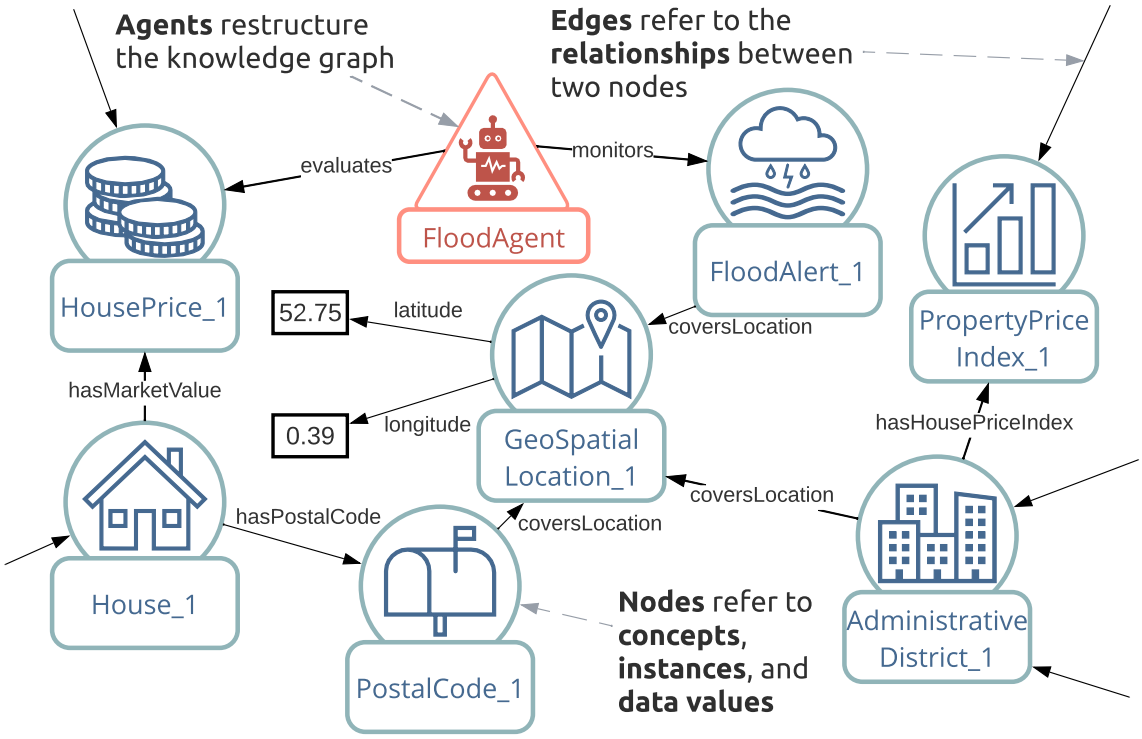


Fig. 2.7 Schematic of the World Avatar KG. Note that the figure is only illustrative and does not reflect actual data.

2.3.1 Vision and status

The term *World Avatar* originated from the idea of the *digital twin* in Industry 4.0 but extended its scope to abstract and represent everything that conceptually and/or physically exists. With this vision, the World Avatar aims to standardise the language used across knowledge domains to enable cross-domain communications, offering extensive opportunities for solving more complex and cross-domain problems [195]. Throughout this process, there is an anticipation to establish connections between individuals and knowledge, aiming to enhance the habitability and sustainability of our environment.

The first instantiation of the World Avatar is applied at the intersection of chemical and electrical engineering [89], with an initial scope of creating a digital replica of the eco-industrial park on Jurong Island, Singapore [271]. The effectiveness of the World Avatar has been demonstrated through its ability to solve many energy-related problems, *e.g.*, the utilisation of waste energy [396], network optimisation of the eco-industrial park [399], and simulations of a carbon tax for scenario analysis in policy making [90], *etc.*

Over the years of development, there is an interwoven network of concepts in the World Avatar spanning temporal and spatial scales, extending from molecule [99] and chemical

mechanism [98] to laboratory [194], cities [45], and even the national level [3]. Based on these ontologies, the physical world we live in is captured and represented as the *base world*. The hypothetical versions of the world in which certain variables or assumptions are different are represented as *parallel worlds*. These alternative universes are managed by software agents that can perform a variety of tasks. Importantly, the World Avatar views these agents to be part of the KG, as depicted in Fig. 2.7.

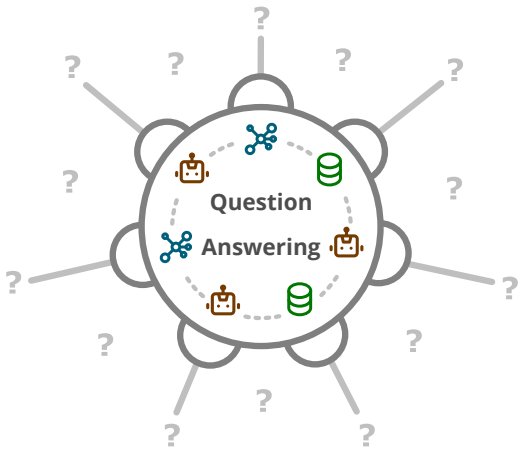
The World Avatar is constantly evolving as it is maintained by a network of active agents who regularly input new data and update existing data. This makes the World Avatar versatile in three aspects, as demonstrated in Fig. 2.8: (1) answering cross-domain questions about the base world [5], (2) controlling real-world entities [159], and (3) supporting what-if scenario analysis with parallel worlds [4, 312].

2.3.2 Design philosophy

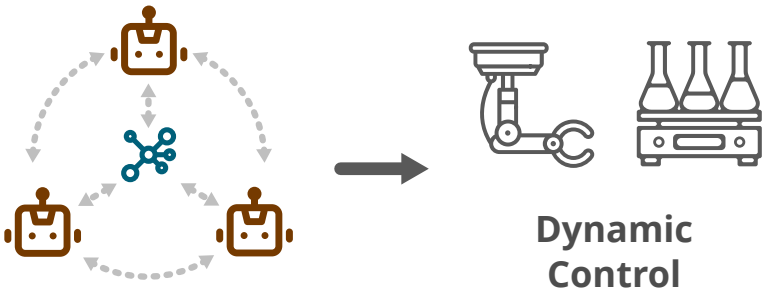
Figure 2.9 presents the main components that contribute to the design philosophy of the World Avatar, *i.e.*, the ontological descriptions of relevant concepts, the data instances, and semantically annotated computational agents that act upon them.

The inclusion of computational agents is enabled by an agent ontology – OntoAgent [401]. This ontology functions as a blueprint for creating interoperable agents, overseeing the concepts associated with their functionalities. Individual atomic agents are endowed with the capacity to perform predefined simple tasks, with their Input/Output (I/O) signature linked to the concepts within domain ontologies. This enabled I/O-based service discoveries to form the agent composition for complex tasks [401]. Once granted access privileges, these tasks are envisaged to be completed without human intervention. Notably, by using OntoAgent to express the agents as part of the KG, the activities of agents are easily trackable so that provenance can be recorded to document the changes in the KG over time. To ensure secure agent operations, blockchain technology was implemented to support automated agent selection with a tamper-proof agent marketplace [402].

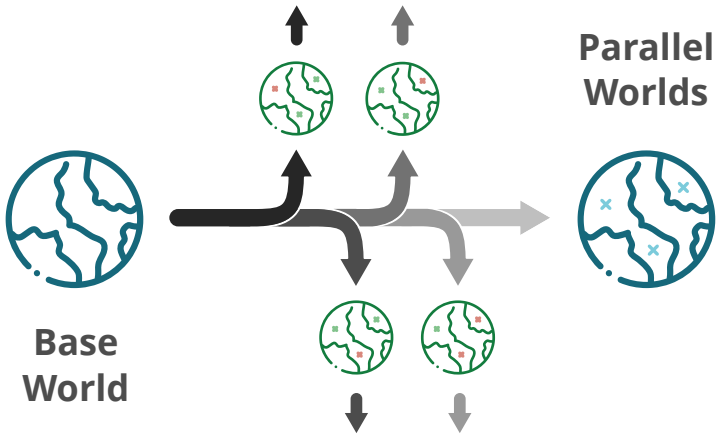
Figure 2.10 describes a three-step plan that is normally followed when developing use cases as part of the World Avatar. It starts from specifying target deliverables to conceptualising relevant concepts and finally implementing codes for queries. In the World Avatar, developing ontologies is often an iterative process and it is not a goal in and of itself [341]. They are typically developed to be digested by software agents which mimic the human way of conducting different tasks [194]. Therefore, the development of ontology typically draws inspiration from relevant software tools and existing database schemas. Views of the domain experts are also consulted to better align with the communal understanding of the subject. During iterations, competency questions are used to test if the ontologies meet case



(a) Capability 1: Answering cross-domain questions about the base world.



(b) Capability 2: Controlling real-world entities.



(c) Capability 3: Supporting what-if scenario analysis with parallel worlds.

Fig. 2.8 Core capabilities of the World Avatar. Used with permission from https://theworldavatar.io/.

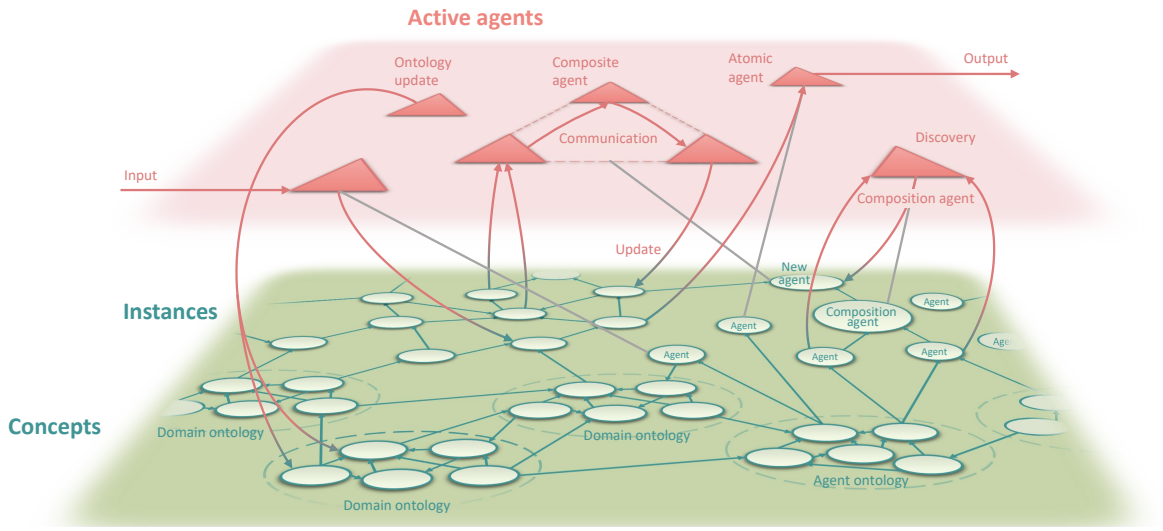


Fig. 2.9 The main components of the World Avatar dynamic knowledge graph. Used with permission from Akroyd et al. [5] © under a CC BY 4.0 licence.

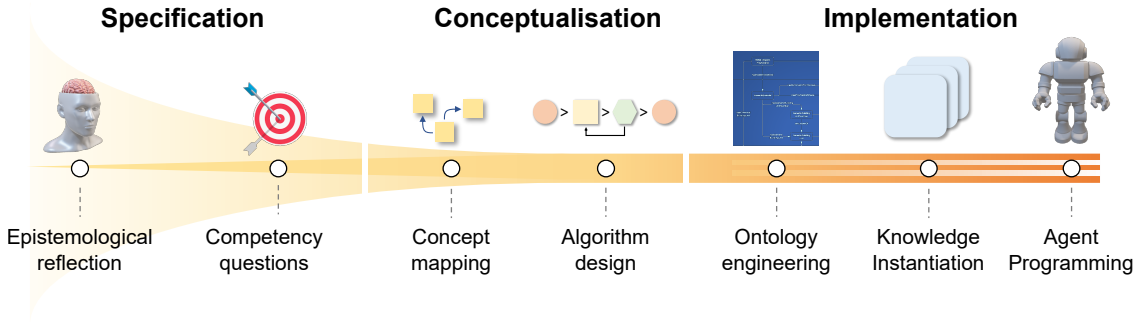


Fig. 2.10 The three main steps followed for a typical use case as part of the World Avatar project. Used with permission from Dr Aleksandar Kondinski.

study requirements Akroyd et al. [5]. The answers to these questions are provided in the form of SPARQL queries that are executed by the agents during their operations. Another essential aspect to consider is data instantiation, where we normally adopted a *persistence layer* to separate agents' internal logic and ontologies' design, simplifying the querying and processing of data from the KG. Overall, the ontology development process starts as easily as drawing concepts and their relationships on a whiteboard and then gradually materialising them in code.

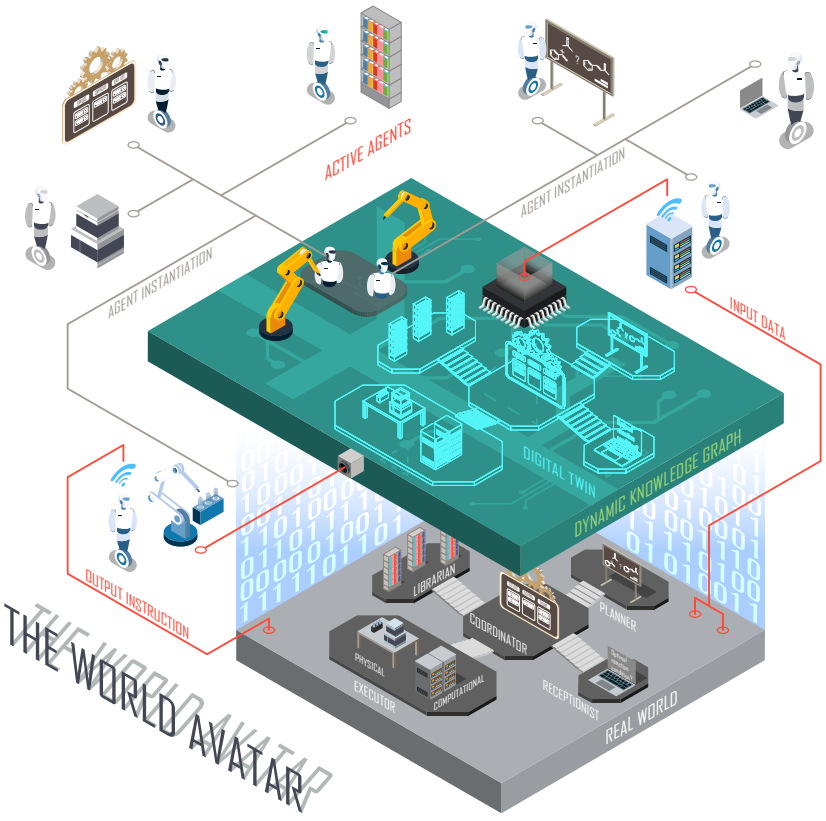
As the World Avatar project continues to evolve, it strives to more accurately represent complex phenomena across spatio-temporal scales. This requires a tool for efficiently coordinating the actions of agents that update and restructure the KG. Currently, software agents are represented similarly to semantic web services [401], which are invoked through Hypertext

Transfer Protocol (HTTP) requests. For time-consuming computations, an asynchronous job watcher is available to delegate jobs to HPC facilities. This was applied to assess the impact of quantum calculations on the air pollution dispersion [248]. These approaches largely adhere to the static remote procedure call paradigm. To fully unlock the potential of the dynamic world model, a more flexible and adaptive architecture that can facilitate autonomous interaction between agents and KG is preferred.

This thesis contributes to the World Avatar from two aspects. Firstly, it presents a generic and robust implementation of an infrastructure designed to manage the knowledge dynamics of agents. This infrastructure facilitates the effective management of agent actions and accurate tracking of their activities. Secondly, the thesis applies this infrastructure within the context of SDLs, showcasing the World Avatar's control capabilities and its expansion from the virtual realm into the physical world.

Chapter 3

Review of community efforts in digitalisation of chemical research



"They need to take the view that data is a precious thing and will last longer than the systems themselves."

– Tim Berners-Lee, interview with Brian Runciman (2006)

This chapter draws from a paper published in *JACS Au* in collaboration with the Computational Modelling Group and the Sustainable Reaction Engineering Group at the University of Cambridge. Contributions from the team included Dr Cao aided in the review and population of Table A.1 and the other authors provided feedback. The remaining review, analysis and writing were performed by the author.

In this chapter, a brief review of contemporary work in the digitalisation of chemical research is provided with the aim of motivating the adoption of dKG technology to address the challenges we identified for establishing a network of distributed SDLs. We begin with reviewing the state-of-the-art implementations of SDLs with a focus on data infrastructure. Based on the limitations of current approaches, we assess community efforts towards standardised data representation and effective data exchange. We identify dKG, i.e., a combination of ontologies and agents, as an interesting technology option. This approach allows the intelligent automation of experiments to be linked with chemical knowledge resources and aligned with other AI techniques. It is suggested that this will play a key role in the next generation of SDLs.

3.1 Platform-based approach

Detailed reviews of the applications of closed-loop optimisation have been published by Cao et al. [42] and Coley et al. [60]. In this thesis, we focus on the data flow between the different components of such an automated experimentation platform as presented in state-of-the-art studies. To have a clearer demonstration of the data flow between different parts, thus revealing how these functional components can be shifted into agents as in the dKG approach, we re-group the five key elements proposed by Flores-Leonar et al. [108] (see Fig. 2.5) and recast them as illustrated in Fig. 3.1. The *receptionist* acts as a human-machine interface that receives, analyses, and translates the requests into machine-understandable objects, as well as enables real-time and interactive communication between the user and data. The *planner* is a decision-making entity that designs the experiment, plans retrosynthesis steps, also selects suitable surrogate models given use cases. The *librarian* is responsible for data management, including maintenance of the database, data cleaning, data validation, and outlier detection. The *executor* performs the computational and physical experiments, both interfaced with the available experimental resources. The *coordinator* manages the entire workflow by locating resources given constraints, requesting data from the librarian, asking the planner for suggestions over the next steps, and requesting experiments from the executor. We categorise the selected studies into the realisation of functional components

and assess the data communication between each of them. It should be noted that we do not cover the specific internal realisation of the components, *i.e.*, we do not consider how the planner handles the input historical data and how it recommends the synthesis route, instead, we focus on the format of the recommendation output from the planner. Following the review, we list the limitations of the platform-based approach which lead to the quest for better data representation and exchange protocols.

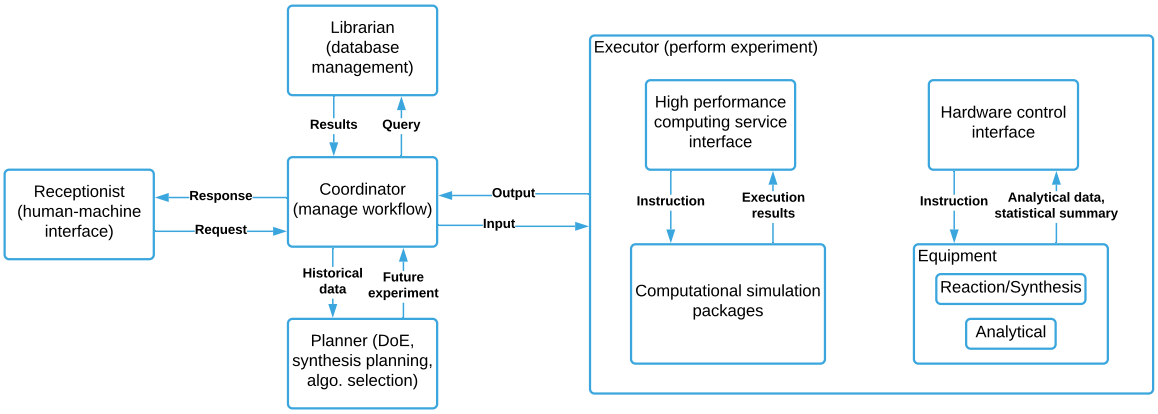


Fig. 3.1 Functional components of a platform-based approach towards chemical discovery, annotated with the communications between each component.

3.1.1 Selected studies

There have been extensive reviews on developing each of the functional components [126, 208, 104, 146, 108]. In the context of chemical automation, Mateos et al. [225] reviewed the realisation of the components in selected continuous flow platforms. In this review, we selected the studies below to illustrate how the data is exchanged between the functional components in the platform-based approach. Specifically, we will review the data exchange protocols between the coordinator, librarian, planner and executor for further investigation on interoperability within one platform and between different platforms in the current setups. We identified three main types of data representation and storage in the automated experimentation platforms, namely, variables stored in a reserved memory location of programming languages, data stored in a file on a hard disk, and data stored in a database. Based on this classification, three types of data transfer and communication protocols were identified as assigning in-memory cache values during software programme run-time, file transfer protocol, and HTTP request/response. It should be noted that although both the latter two ways of communication belong to the application layer in the TCP/IP model, they are distinguished

herein to emphasise the format in which the data is stored and consequently transferred. To the best of our knowledge, the complete details are summarised in Appendix A.

Receptionist

The receptionist acts as the human-machine interface. Among different platforms, multiple ways of interaction have been reported. Knight et al. [192] present a voice-controlled user interface integrating voice, text, and visual dashboards. This increased the flexibility for the experimentalist to communicate and collaborate with the automated setups without the coding experience required. Web interfaces via HTTP requests/responses [106, 105, 172] is another way of interaction. The advantage of this approach is that authorised users can log in to the web page and access the platform from all over the world [104]. Moreover, the Natural Language Processing (NLP) modules can build on top of the web interface as chatbots, which can further connect to existing messaging services such as Gmail, Twitter, Slack, and Dropbox [303, 221]. The Graphical User Interface (GUI) is a more intuitive way of interaction between the users and the automated experimental platforms. It can be built through different coding software, such as Matlab [178], Python [347, 209], and LabVIEW [17, 238, 50]. It should be noted that each receptionist can only work within its own operating system due to its bonded communication protocols as well as the coding language.

Coordinator

The coordinator manages the workflow in the closed-loop system. Among the different programming languages/tools that have been employed to develop the coordinator, Python is perhaps the most widely adopted. The Aspuru-Guzik Group proposed ChemOS [303, 221], a modular coordinator orchestrating the learning module (the AI-based planner), the communication module (server-based receptionist) and an operation module for remote control of the robotic platform. ChemOS demonstrated decision-making capabilities in managing the workflow for thin-film material discovery [221] and increasing the efficiency of organic photovoltaics [203]. It has now been commercialised as Atinary SDLabs [9] as a no-code platform. Zhu's group presented the Materials Acceleration Operating System In Cloud (MAOSIC) [209], a coordinator in cloud upgraded from their previous system MAOS [210], which was applied to the autonomous discovery of optically active chiral inorganic perovskite nanocrystals. Experiment Specification, Capture and Laboratory Automation Technology (ESCALATE) has a coordinator acting as a bridge to connect the experimental workflow [277]. Its initial implementation was designed for the exploratory

synthesis of single-crystal metal-halide perovskites. Further discovery of the formation of two new perovskite phases was demonstrated [212]. Chemputer [347] was developed for organic synthesis optimisation in batch reactors. This coordinator brought together synthesis abstraction, chemical programming and hardware control, and tested the synthesis of three small pharmaceutical compounds with similar yields to those obtained by manual work. Moreover, by using a standardised format for reporting a chemical synthesis procedure within the coordinator, Chemputer captures synthetic protocols as digital code that can be further published, versioned and transferred flexibly [143].

LeyLab [105] is a PHP-based coordinator orchestrating multiple users and equipment in different continents for the development of catalysts and process conditions in flow reactors. The firewall within the coordinator prevents malicious attacks from unauthorised users.

The Lapkin group presented a Matlab-based coordinator for multi-objective optimisation of the reaction conditions for SNAr and N-benzylation reactions [323]. It demonstrated its flexibility to a different chemical system with an aldol condensation reaction optimisation [178].

There are also coordinators based on LabVIEW. Given the user-friendly graphical programming interface in LabVIEW, building a receptionist module is not required in this setup. However, Matlab [238] or Python [50] are occasionally paired up with LabVIEW to enable the planner module to suggest new experiments.

Another notable development is C#-based ARES OS [2], an open-source software released by Air Force Research Laboratory (AFRL) following their Autonomous REsearch System (ARES). As the first reported autonomous experimentation system for materials development, ARES demonstrated its capability in carbon nanotube synthesis experiments [258, 259], and additive manufacturing applications [79].

It can be seen that coordinators followed different coding philosophies in different programming languages. For each case study, the reported coordinator indeed satisfied the specific need yet fail to extend to other systems.

Coordinator – Librarian

The interaction between the coordinator and librarian focuses on reading historical data and writing new data for data storage. Depending on the operating system of the coordinator, as well as the structure of the librarian in each platform, the data communication protocols between the coordinator and librarian are various.

An intuitive approach is to store and transfer the data as variables in the memory of the operating system. Jeraal et al. [178] stored and transferred data as Matlab variables. Similarly, Christensen et al. [53] used Python variables for communication. This approach

is lightweight and independent of the database structure. However, it is vulnerable as there is no backup for the data obtained. Moreover, the data stored are hard-coded and picked beforehand, meaning the variables will be reassigned during the iterations.

File transfer is an approach to overcome this issue. Cao et al. [42, 41] used CSV files as the bridge for communication. Other studies used MAT files in a similar fashion [381, 17]. In this approach, the experimental results were exported and stored as a file that can be loaded later for suggesting the next experiments. Compared to storing data as in-memory cache variables, the file transfer approach gives a way to back up the data on a separate machine or online server with flexible access and secure storage. However, the files can still be hard to track and classify when the number of experiments is high or more than one type of experiment is run on the platform.

Databases provide a solution to efficiently manage large amounts of experimental data. Li et al. [209] stored long-term data through SQLAlchemy which supports a DataBase Management System (DBMS), with databases such as MySQL, Postgres, Oracle, and SQLite as the back-end. The coordinator MAOSIC can read and write new entries to the server-based database via API. In Roch et al. [303], the coordinator ChemOS was connected to SQLite, and the information was stored in four distinct databases (requestDB, parameterDB, robotDB, feedbackDB) on SQLite to better classify the data and retrieve them in the later stage. Materials Experiment and Analysis Database (MEAD) [338] consists of both raw data and metadata from high-throughput experimentation. By instantiating an Event-Sourced Architecture for Materials Provenances (ESAMP) [343], the MEAD database enabled the ML algorithm to utilise the material state within its experimental workflow for accelerating materials discovery.

Coordinator – Planner

To avoid an exhaustive search of the chemical space, the planner needs to decide which new experiments should be conducted. Depending on the purpose of the platform, the planner algorithm can be classified into discovery and optimisation. Detailed reviews of the existing algorithms for planner have already been published; interested reviewers refer to Garud et al. [120] and Clayton et al. [55]. The communication between the coordinator and the planner is mainly done in two ways: variable stored in memory [43, 17, 221], and file transfer [41, 323, 347, 59]. It is worth mentioning that the communication protocols are not necessarily the same over one platform. Li et al. [209] used database queries for the interaction between the coordinator and librarian, yet they depend on Python variables for the communication between the coordinator and planner. It can be seen that the platform-

based approach can adapt to different ways of data exchange, yet modifications that are case-sensitive will be needed.

Coordinator – Executor

The executor runs the experiments, computationally or physically, and sends back the experimental results. The interaction between the coordinator and executor module highly depends on the operating system for the instrument, as the actual experiment resources within the executor are normally surrounded by a layer of interface. Therefore we review the communication protocols of the physical and computational experimental platforms separately.

Physical experiment interface Robotic platforms have their origins in instances such as peptide synthesis [233] and the pharmaceutical industry [385, 214]. Some existing commercially available semi- and fully-automated platforms in chemistry have emerged as powerful tools and can be embedded into the closed-loop optimisation system [108].

Commercial platforms provide various high-throughput workflow solutions, ranging from single bench-top/standalone automated workstations up to complete and integrated product development workflows for the entire product development processes in chemical material science [227, 163]. Greenaway et al. [133] applied the Chemspeed Accelerator SLT-100 synthesiser platform in the discovery of porous organic cages and the optimisation of the cage formation conditions. This platform can carry out up to 96 reactions in parallel, highly speeding up the testing of the proposed experimental conditions that are sent to the platform via file transferring within the Chemspeed custom software. The hardware from Chemspeed is also used by IBM's RoboRxn [270], a remotely-accessible automated organic synthesis platform utilising various Transformer [386]-based ML algorithms for chemical reaction prediction [320], retrosynthetic pathway planning [321], synthesis action extraction [369], and chemistry grammar extraction [322]. Vapourtec delivers an automated flow reaction platform with multiple choices for pumps, and flow reactors. Successful examples of using the Vapourtec system in the closed-loop optimisation setup include drug discovery [121], scale-up development [370], and reaction condition optimisation [323, 178]. It is worth mentioning that commercially available mobile robots and robotic arms have been used in complex and multi-step operations [35, 59]. Communication between the coordinator and the robots was achieved using various communication protocols: Transmission Control Protocol/Internet Protocol (TCP/IP) over Wireless Fidelity/Local Area Network (WiFi/LAN), RS-232, WebSocket, *etc.* Although commercial systems developed by various vendors are easily implemented with a user-friendly user interface, it limits the experimental choice

across platforms, and it is hard to configure the platform to the existing workflow architecture and setups in the lab.

To enable a modular-based plug-and-play platform, single-board controllers, *e.g.*, Raspberry Pi and Arduino, were used to act as the interface layer connecting the coordinator to the actual experiment executor, *i.e.*, sample preparation, analytics *etc.* This is favoured by the academic community due to its flexibility and compatibility with different experimental instruments at a relatively low cost. The communication protocols between the coordinator, single-board controller and experiment executors are various. A TCP/IP protocol was used when a Raspberry Pi was applied. Fitzpatrick et al. [106] used a Virtual Local Area Network (VLAN) to control lab equipment, also a Secure Shell (SSH) tunnel between the virtual environment and the remote control server. Similarly, Roch et al. [302] controlled the pump system using the Raspberry Pi and interacted via a Secure Copy Protocol (SCP) with the executor codes. In Chemputer designed by Steiner et al. [347], an Arduino was designed as the microcontroller. Instances of experiment executors are created as Python instances at the initialisation stage and the coordinator reads related information stored in a GraphML file. Li et al. [209] conducted their high-throughput experiments via an Arduino control board as well but followed the JSON Remote Procedure Call (JSON-RPC) 2.0 protocol used for robots and characterisation equipment control. A detailed review of microcontrollers and their applications in automated experimental systems can be found in Fitzpatrick et al. [107]. The in-house built platform can connect to different lab equipment based on the users' needs and existing lab setup, yet different communication protocols prevent it from extending to other labs/systems.

Robot Operating System (ROS) [287] is the *de facto* standard middleware in the robotics field for orchestrating multi-robot systems. In 2019, Marquez-Gamez and Maffetton [224] proposed a ROS architecture for laboratory robotics motivated by Burger et al. [35], envisaging a "cobot" future where human researchers and robots work collaboratively in the chemistry lab using modular and reconfigurable lab equipment interfaced via ROS. A recent paper from Fakhruldeen et al. [96] shows proof-of-concept towards this direction.

Computational experiment interface With the rapid development of computational power and simulation methods, computational experiments are playing a more vital role in catalyst design and optimisation [368], synthesis planning [355] and catalyst discovery [362]. By using theoretical, fully automated screening methods combining ML and optimisation to guide Density Functional Theory (DFT) calculations, Tran and Ulissi [361] screened across intermetallics for the discovery of electrocatalysts for CO_2 reduction and H_2.

The main executor for computational experiments is the HPC. However, the interaction between the HPC and the coordinator on local computers is different from case to case. The scheduler is the interface for the users on the login nodes to submit batch jobs to the compute nodes on the HPC, as the users cannot run their calculations directly and interactively (as they do on their personal workstations or laptops). The scheduler stores the batch jobs, evaluates their resource requirements and priorities, and distributes the jobs to suitable compute nodes.

There are quite a few open-source scheduling software depending on the setup of HPC, among which Simple Linux Utility for Resource Management (SLURM) is widely used in research computing services [297]. Rosen et al. [306] developed the PyMOFScreen Python package to manage automated DFT calculations, leading to new electronic structure database constructions and accelerating new materials discovery [305]. Multiple software packages were developed to enable high-throughput screening on the HPC, such as Python Materials Genomics (pymatgen) [265], FireWorks [177], custodian [265], Atomate [226], GASpy [361, 362], and ChemEco [140, 141]. Depending on the user's need as well as the DFT calculation software, the structure and the output file of those Python packages are different and non-transferable. A notable effort in addressing this issue is MolSSI QCArchive [240], which offers open access to millions of quantum chemistry calculations done with different software, as well as on-demand computation.

3.1.2 Current limitations

Despite the huge improvements made in the literature, a few limitations remain to be overcome before it is possible to achieve a global collaborative network [342]. The platform-based approach presented heavily relies on the coordinator. This increases the possibility of data loss during transmission, and it will become unsustainable soon with further expansion of the ecosystem. Direct communication between functional components is one potential approach to mitigate this issue, as demonstrated by Fitzpatrick et al. [106] in letting the planner directly communicate with lab equipment via TCP/IP.

Another limitation is the *ad hoc* data representation and storage. This is particularly important as there is no standard method of representing results or recipes for chemical experiments, despite several competing standards of representing molecules co-exist. The heterogeneous data format lacks interoperability which precludes the full utilisation of the embedded information. This problem is further exacerbated when the collaboration between different groups is considered; potentially data generated from one group will be shared and tested on the platform of another group for reproducibility and further experimentation. Moreover, the consequent various data transfer and communication protocols result in low

extensibility issues as a considerable amount of time is often required when new hardware or software is integrated, also noted by Breen et al. [32].

Unbalanced chemical data is another limitation to be addressed [61]. In ML applications, historical data from reaction databases are normally applied as the training set to guide the learning of the planner models. However, only "good" experiment results are published and stored in these databases, limiting the opportunity of learning from "bad" examples [288]. When coupled with the limitation of *ad hoc* data representations, this leads to a situation where many platforms generate experimental data from scratch without utilising the prior chemical knowledge. A further issue lies in several examples where users are required to manually input chemical data [335, 178]. This is error-prone and limits the potential of full automation.

In brief, improving the interoperability within one platform and between different platforms is a key step in lowering the entry barrier of digitalising chemistry and promoting a fully connected network of SDLs. It is thus important for us, as a community, to know how far we are from meeting the prerequisite condition – a fully interconnected data representation capturing the data generated within the experimentation.

3.2 Data representation and exchange protocols

As promoted by various researchers [156, 146, 61, 382], the digitalisation of chemistry facilitates the collaboration between research groups. Figure 3.2 reviews data representation and exchange from the different perspectives of a chemical experiment, namely, molecule, reaction, analytical data and method, procedure and hardware, and finally holistic data capture and exchange. Importantly, we distinguish the community efforts into non-semantic and semantic paradigms depending on whether chemical ontologies are involved and lay out the connection between them. The agent-based approaches towards standardised and effective communication between each of the components involved are discussed.

3.2.1 Non-semantic representation

In this review, we broadly distinguish non-semantic efforts into four parts: a representation of cheminformatics formats, a schema for constrained encoding of data, a collection of data stored in a database, and finally a holistic architecture that aims to capture all data generated within an experiment.

Since the discovery of the periodic table of elements, chemical knowledge is built on structures with competing representations [400]. The most commonly used representation

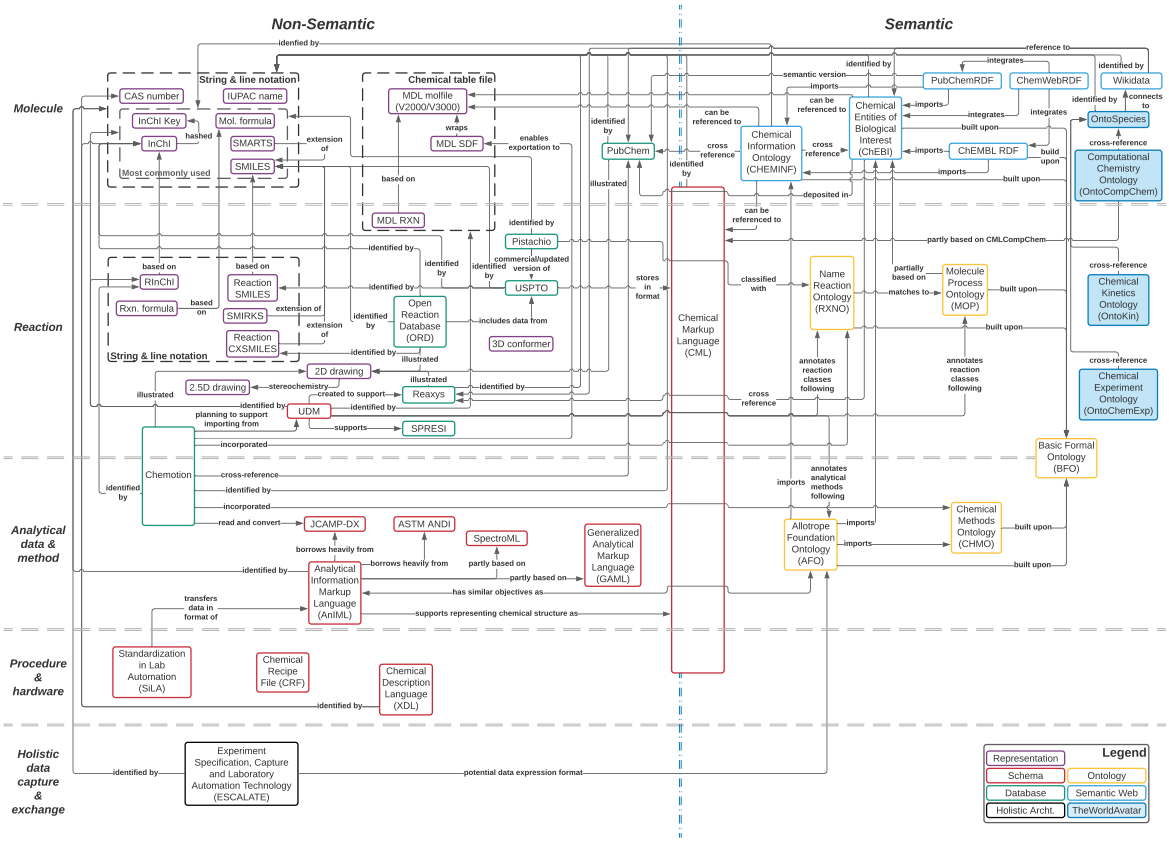


Fig. 3.2 The community landscape towards a better data representation and exchange in chemical digitalisation. The focus of each category: (a) molecule: chemical structure, physicochemical properties, spectral information of a given species; (b) reaction: chemical reaction scheme, conditions, description of procedures, and statistic summary of the reaction outcome; (c) analytical data & method: analytical data collected and the methods applied within the experimentation, this is distinct from the spectral information of a given species as this focuses on the data collection process; (d) procedure & hardware: the operational procedure in an experiment in the format that can be directly executed by hardware; (e) holistic data capture & exchange: the initiatives to capture all the experimental information generated within the experiment and the exchange of data between different hardware/software. For those on the fence between two categories, we meant they cover both areas. Chemical Markup Language (CML) was labelled as both semantic and non-semantic since it preserves hard-coded and rule-based semantics but not ontologies following semantic web standards [374]. Basic Formal Ontology (BFO) is an upper-level ontology as the basis of other ontologies and it does not capture any domain-specific information.

is string and line notation, including SMILES [380], InChI [153], SMARTS [68], SELFIES [199], *etc.* for molecules, and RInChI [134], SMIRKS [69], *etc.* for reactions. Chemical table files express molecules and reactions in terms of *x*-*y*-*z* coordinates of atoms and bonds. For a more visual representation, molecules and reactions can be illustrated with 2D line drawings (or 2.5D including stereochemistry), and 3D conformers. These formats are interchangeable with the help of cheminformatics tools, *e.g.*, Open Babel [262] and RDKit [202]. An ML application normally starts with encoding structural representations in the form of high-dimensional vectors to map the implicit chemistry to either the physicochemical properties of one molecule or the reactivity between different molecules.

Popular chemical databases and registry systems normally store various representations of the above with registry numbers, *e.g.*, IUPAC name, CAS number and PubChem CID, for unique and unambiguous identification within themselves and cross-reference between repositories. PubChem [187] is the largest open-source structural chemical information repository. For reaction informatics [257], the scale of open-source databases is much smaller. The United States Patent and Trademark Office (USPTO) database [219] is one of the seminal databases in the community that contains 3.7 million reactions extracted from US patents. It was commercialised as Pistachio [256] containing more than 13 million reactions with annotated reaction classifications using named reaction ontology (RXNO [316]) and expanded coverage to other patent offices, *i.e.*, World Intellectual Property Organisation (WIPO) and European Patent Office (EPO). Despite the public availability of the USPTO database, its representation schema, *i.e.*, Chemical Markup Language (CML) in XML, requires extra efforts of format transferring for ML applications. This results in different versions of the USPTO subset that were derived and adapted by various researchers for their applications [320, 28, 179, 319]. As the tailored database can be kept private to the research group, it could be difficult for bench-marking new algorithms.

To facilitate the development of ML in chemistry, Open Reaction Database (ORD) [268, 184] was formed to encourage pre-competitive data sharing in a standardised format. It records how the reaction was performed, including reaction inputs, conditions, outcome, *etc*. Notably, ORD uses a protocol buffer as its data structure, instead of the commonly used XML schema. It deliberately avoids the use of ontologies due to insufficient ML applications with ontologies seen in the community [183]. Despite ORD storing the operation sequence in a machine-readable format, the authors declared it a non-goal at present to make it compatible with programmatic execution on automated synthesis hardware. For more complex operations, ORD only supports a free-text description of the procedure. In terms of the reaction outcome, it focuses more on the statistical summary of the reaction, *e.g.*, conversion and yield, and

unprocessed analytical data if available. At present, ORD contains 2 million reactions [183], including part of the USPTO dataset that was converted from CML.

Unified Data Model (UDM) [280] is another initiative aiming at capturing and integrating the experimental information generated during chemical synthesis. UDM was originally developed by Roche as a transfer model of MDL RD file format for integrating data from various sources into Reaxys database [130]. It has since evolved to an XML schema with three main elements, namely, *citations*, *molecules* and *reactions*. In addition to recording the molecule and reaction identifiers, UDM annotates its data with semantic vocabularies. The reaction classification is based on the molecular processes (MOP [92]) and RXNO ontologies, demonstrated by its sample data taken from Reaxys. The analytical method and results type are based on a working draft version of Allotrope Foundation Ontology (AFO [235]) where duplicate entries exist. However, it should be noted that the way UDM integrates the ontologies is by enumerating the ontological classes as a sub-schema of UDM and tagging them to the XML elements as attributes. One general issue with this type of enumeration and attribution is that the relationships declared in the ontologies are not retained in the XML schema, *e.g.*, class and subclass relationship between concepts in MOP and RXNO, and the corresponding relationship between result types and analytical methods in the AFO. By looking at the publicly available resources, there are no programmatic constraints over how ontological axioms are enforced in a UDM file. Moreover, UDM allows any type of format for the analytical data recording, at least by XML schema itself, tailored tools would be necessary for better utilisation of the data. In its latest release, UDM extends its support to the SPRESI database [307]. Moving forward, UDM aims to provide fully captured representations of reaction predictions and optimisations for multi-step reactions. Additional support for environmental health and safety data is also of interest [23].

Similar to ORD, Chemotion [363] aims to build a community-driven repository to better publish reaction data generated across different laboratories. In practice, despite containing fewer data, a key distinguisher of Chemotion is its level of interoperability in enabling programmatic transfer of raw analytical measurements for integration of ELN from individual laboratories. It does so by supporting reading and converting analytical data in the widely-used JCAMP-DX format [201]. Each published reaction in Chemotion has a semi-machine-readable format with a Digital Object Identifier (DOI). It cross-references compound entries in PubChem. Like UDM, Chemotion incorporates ontologies (RXNO and chemical methods (CHMO [93])) for semantic annotations at a vocabulary level. On the data validation front, Chemotion automates the curation of some types of analytical data, *e.g.*, plausibility checks of Nuclear Magnetic Resonance (NMR) data. Human inputs are still required to ensure data quality for publication. To enable more data resources, Chemotion

is planning to support reactions stored in a UDM format. Chemotion is also planning to connect ELN to robotics to establish an automated platform for chemical synthesis [180].

As mentioned, JCAMP-DX is a data standard widely used for recording and sharing analytical data. However, one drawback to its utilisation is the lack of validation tools making it difficult for data generated from different software to adhere to the standard terms [360]. One approach to alleviate this problem is modernising the standard terms with an XML schema, such as Analytical Information Markup Language (AnIML) [7]. AnIML is partly based on SpectroML [308] and Generalised Analytical Markup Language (GAML) [360], also drawn from JCAMP-DX and ASTM ANDI. On the chemical structure side, AnIML supports the CML format together with other commonly used line notations. AnIML aims to provide vendor-neutral analytical and biological data representations that are designed for manufacturers to install and maintain. For the same reason, AnIML provides audit trials and other metadata for reporting information in regulatory processes. At present, AnIML supports the most common analytical equipment with detailed documentation for Ultraviolet–Visible spectrophotometry (UV/Vis), chromatography, and indexing.

Up to this point, reviewed efforts are standardising the data generated during the experiment. Initiatives exist to standardise the instrumentation interface, *e.g.*, SiLA [332]. SiLA is a micro-service architecture using Google Remote Procedure Call (gRPC) and HTTP/2 protocols with a protocol buffer as its payload. It adopts a client/server view to describe the devices in the lab environment, where entities expose (multiple) services as SiLA Features accessible to others. SiLA Features are expressed in a predefined XML-based schema and stored in an online repository for service discovery. Each feature is assigned a unique identifier to enable peer-to-peer interactive communication, status queries, and reactions to events. As SiLA is a communication protocol for equipment control, it utilises AnIML as the medium for the bidirectional transfer of analytical data between Laboratory Information Management Systems (LIMS) and Chromatography Data Systems (CDS) in a file-less fashion [313]. The combination of SiLA and AnIML represents a promising direction: standardised interfaces for instrumentation and unified machine-readable data representations. This results in a complete data package after completion of the analytical experiment, including all the process steps and the generated data.

Whilst SiLA standardises equipment interface, Chemical Recipe File (CRF) [59] and Chemical Description Language (χDL) [229] are initiatives to automate experiment execution. They both focus on translating the operational procedures from unstructured descriptions to robot execution commands.

CRF [59] is a CSV-based schema developed for flow synthesis. Since the instructions are generated based on batch reaction data, human modification is required to enable continuous

processes. One notable aspect of their setup is their modularised reaction hardware, making it robotically self-reconfigurable, as demonstrated by the back-to-back synthesis of medicinally relevant small molecules.

χDL [229] is an XML schema focusing on batch synthesis. It contains three main components as the apparatus to be employed and manually configured, chemicals to be used, and robotic steps abstracted from operations used by chemists in the lab. An ontology is proposed to map the command and hardware executions, however, it is not published in semantic web standards [374]. Before the instructions are sent to execution, researchers can modify the conditions to benefit human intuitions.

Both CRF and χDL focused on providing a flexible framework to conduct synthesis for multiple molecules. However, neither of them included an automated analysis step. The statistic summary of the chemical synthesis is thus not provided in a standardised format as done by other reaction schemas.

ESCALATE is an attempt towards holistic data capture and exchange [277]. It proposed an ontological framework for experimentation, supporting data collection, reporting and experiment generation. This framework captures and reports all the reactions conducted, including "bad reactions" – in line with the cultural change promoted by the community [156]. In its first release [277], the claimed ontological framework was realised by implementing template-based files to store the experimental information, *e.g.*, CSV and text files in a file-sharing folder infrastructure (Google Drive). The authors additionally acknowledge that the Allotrope Foundation Data Standard could be incorporated into this data lake. Despite Uniform Resource Locators (URLs) being employed as pointers to some data, the data representation remains heterogeneous and only semi-structured, without the semantic features required by semantic web standards [374]. In a more recent development [260], an ESCALATE REST API [95] was made available to showcase the possibility of retrieving chemical informatics data from PubChem API, interacting with a Postgres database for submitting experiment jobs to a laboratory and querying the hosted results.

In general, the non-semantic efforts are closely connected to each other. Multiple representations are normally used within schemas or databases to meet the needs of different applications. Databases cross-reference to each other using registry numbers.

Another notable trend is the adoption of XML schema as data structures. XML is a machine-readable format for algorithmic operations. It relies on string parsing when automating some of the processing steps. For example, the automated unit conversion provided by χDL, where the case-insensitive conversion to a standard unit was performed. However, XML is not designed to host large sets of data as querying between different files can be challenging. The linkage between entries in XML is implicit and requires tailored

codes to handle. A solution to this problem could be hosting data in a database and exposing that as the query interface. Yet as demonstrated in the platform-based approach, the same scalability issue would emerge.

It is worth noting the efforts to improve interoperability. Most of the schemas classify items using annotations based on ontological taxonomies. There are also works that claim to have developed ontologies, but that are not however represented in a formal ontology language such as OWL – their data is still file-based. In the context of this thesis, we consider these outputs to be taxonomies that formalise the hierarchical relationships, distinguishing them from the chemical ontologies that are introduced in the next section. The difficulty of achieving general interoperability remains an issue to be addressed.

3.2.2 Semantic representation

Following the general introduction of the semantic web technologies in Chapter 2, the focus herein is the uptake of such technologies in the chemistry domain, as illustrated in the right half of Fig. 3.2. For initiatives where only TBox are available, we labelled them as "Ontology", whereas ABox are published are labelled "Semantic Web". Those under "TheWorldAvatar" refer to the ontologies developed in-house.

Chemical informatics has a long history of utilising semantic web technologies. The chemical semantic web [123, 253, 58] is one of such early attempts by Murray-Rust and co-workers, contemporaneously to Berners-Lee's proposal of the semantic web [21]. In their work, CML was employed to host the data, prior to OWL becoming the semantic web standard. CML schema covers concepts related to atoms, molecules, computational chemistry, crystallography, spectra, chemical reactions, and polymers. It greatly influenced the development of reaction informatics, especially, it is the molecule representation implicitly used by various cheminformatics software [251].

Since OWL became more and more popular in modelling ontologies, more activities of ontology development have been demonstrated in the scientific domain. Despite the authors of CML holding the view that ontologies following the semantic web standards [374] are "too complex for the chemical community to take on board, and provides little effective added value" [252] compared to their approach, the benefit of semantics motivated the development of chemical ontologies to a great extent, especially work at Royal Society of Chemistry (RSC) [15], *i.e.*, CHMO [93], RXNO [94], and MOP [92]. These ontologies are sophisticated and carefully curated. As demonstrated in the non-semantic efforts, they are widely-used for annotating reaction classes and analytical methods.

Another driving force of ontology development in the chemistry and biology domain is the European Molecular Biology Laboratory's European Bioinformatics Institute (EMBL-EBI).

In contrast to RSC ontologies that only provide concepts, EBI ontologies provide knowledge at both a terminological and assertional level, covering small molecules (ChEBI [148]) and cheminformatics (CHEMINF [147]) in a cross-referenced fashion. CHEMINF supports molecular structure representations in the CML format, it also partly transformed data from PubChem into a knowledge base together with cross-reference to their PubChem entries. ChEBI deposited its data in PubChem entries and cross-referenced to Reaxys entries. These ontologies complement other ontologies in the field. For example, CHMO intends to describe the physical and practical methods, whereas CHEMINF covers the computational and theoretical ones.

Ontologising existing databases was demonstrated in the community, including ChEMBL RDF [384] and PubChemRDF [113], the semantic version of the current largest open-source chemical information repository – PubChem [187]. However, the RDF version of these databases did not come with an officially supported SPARQL endpoint. Galgonek and Vondrášek [114] recently addressed this issue by integrating PubChem, ChEMBL and ChEBI datasets as a PostgreSQL database and exposing that to support SPARQL queries. This enabled fast access to chemical data from different sources.

The Allotrope Foundation is a collaborative effort from the pharmaceutical industry [235]. Similar to AnIML, it aims to propose a common data exchange format to unify the laboratory Information Technology (IT) landscape. It started from realising the vision of Roberts et al. [300, 301] where an XML schema was envisaged to provide a holistic data format. It later decided to store data based on HDF5 and RDF formats that were controlled by ontologies for semantic capabilities. The foundation now contains three ontologies, namely, AFO, Allotrope Data Format (ADF), and Allotrope Data Model (ADM). AFO is the ontology at the TBox level representing the knowledge in the chemistry domain and it borrows heavily from CHMO. ADF refers to the ontology ABox classified by AFO, extended with more features on data structure and provenance for long-term archiving. ADM is the constraint for how data in ADF should be modelled following AFO. However, only AFO is freely accessible to the public, with the remaining resources restricted to community members.

Compared to non-semantic efforts, a key distinguishing factor of the semantic approach is its fully-linked concepts and data instances. This is particularly true for the ontologies reviewed above, as their concepts follow the classification of the Basic Formal Ontology (BFO). The instances stored under each ontology are inherently linked and consistent in logic. This enables interoperability between domains and easy access to data from different sources via SPARQL queries. Moreover, the linked nature made it possible to reduce duplication of information by providing unique identification to the entities, whereas in XML it would be

more likely that the same information would appear in different files, *e.g.*, when the same molecules are involved in different reactions.

The biology community has demonstrated the population of data is the key to a broader impact with well-defined ontologies [13]. However, classifying and annotating data into ontologies while maintaining logical consistency is a challenging task, especially with complex ontologies. It is costly to adopt and creates a high entry barrier. This is reflected in reaction informatics, as ontological data is still very much limited to chemical species information, and there is currently no semantic version of reaction data available. This further exacerbated the problem of insufficient adoption of semantic web technologies in ML and other practical engineering applications, as noted by the developers of ORD [183]. Not to mention to actually control the equipment execution and automate the data exchange framework is even more challenging. A trade-off between engineering practices and comprehensive representation is thus important. A potential solution to this would be to convert existing databases into RDF [230].

The same issue was acknowledged by the Allotrope Foundation [235] that there is a trend of making simpler data models for practical applications. One of their partner companies, TetraScience, developed an Intermediate Data Schema (IDS) – a JSON-based schema of analytical data as the precursor of the AFO format. Using an agent, data generated from the analytical equipment was collected and converted to ADF for further analytics. Despite of being proprietary, it enlightens the way forward to standardise data conversion and integration while it is generated. A perspective from Godfrey et al. [127] backed this idea, *i.e.*, data stored in an ontological framework would very much facilitate the proliferation of interoperable standards, also keep the flexibility of introducing new methodologies.

The World Avatar provides ontologies to describe a variety of concepts in chemistry. For example, we have developed OntoKin [98] as an ontology for representing chemical kinetic reaction models and OntoCompChem [197], based on CML [279], to store quantum chemistry calculations. We further introduced OntoSpecies [99, 273] for unambiguous identification of the chemicals, allowing for consistency checking across multiple mechanisms [100] and seamless translation of chemical names when integrating chemical data gathered from different sources. The ontologies are connected to many of those developed by the community. For instance, the development of OntoCompChem is partly based on the CompChem terms as described in the CML and the Gainesville Core (GNVC) ontology [51]. The relationship between these ontologies and other data representations used by the community is shown in Fig. 3.2.

3.2.3 Agent-based approaches

In the context of SDLs, agent-based approaches can be adapted to replace the functional components within a platform-based approach. Montoya et al. [241] wrapped different algorithms as agents to suggest the next experiments for DFT calculations on stable materials discovery. Gomes et al. [129] standardised various tasks as agents (bots) in a platform for crystal-structure phase mapping. Caramelli et al. [43] applied agent-based model simulations to showcase the effectiveness of multi-threaded networking principles in searching for the optimal solution in the chemical space.

In the above studies, a step was made to turn functional components into modularised agents and standardise the data exchange between them. However, the communication was done by passing in-memory programming variables [129, 241], or posting plain text on a human messaging platform (Twitter) [43]. As discussed in earlier sections, the same drawbacks such as lack of scalability and interoperability will emerge when forming the network of SDLs. A relevant first step towards addressing this issue is demonstrated by DLHub [48], which allows users to publish, share, and cite ML models for applications in science.

3.3 Dynamic knowledge graph approach

In this section, we explore how a combination of semantic web technologies and multi-agent systems – a dKG approach – might be applied to realise a complete digital and self-driving laboratory, *i.e.*, a chemical digital twin. Following the introduction of the World Avatar project in Chapter 2, we herein outline a conceptual example of automated closed-loop optimisation powered by dKG and assess its potential to achieve full automation.

3.3.1 Knowledge graph value proposition

A core strength of dKG is interoperability as it provides a mechanism to combine data, descriptions of software, hardware interfaces and experimental workflow in a standardised way, facilitating automation and allowing communication between agents acting on data from different domains [248].

Another key feature is the open-world assumption which enables the scalability of a dKG system. Once the skeleton ontology is set, extending knowledge coverage and tailoring against specific applications is easy to manage. It should work just like adding new features to a computational library.

Moreover, once the code of conduct is defined for each agent, they can act autonomously and modify the KG as time elapses. By doing so, dKG reflects and influences the ever-evolving status of the real world.

3.3.2 Automated closed-loop optimisation

The characteristics of dKG open up the possibility of a new and powerful approach to closed-loop optimisation. In this section, we explore how to apply dKG technology to do this in the context of a case study that was previously automated using a platform-based approach [178]. The case study considers flow chemistry. However, given suitable ontologies and agents, the underlying principles are expected to be generalised to any practices in chemistry where a DMTA loop is involved.

Figure 3.3 illustrates the whole framework consisting of three layers, namely, the real world, the dynamic knowledge graph, and active agents. Reaction data are expressed in ontologies and hosted in the dKG, together with the digital twin of the lab equipment and interoperable agents. Once activated, these agents act autonomously over the dKG and keep the cyber- and the real-world synchronised. The update of the digital twin is based on the readings from the equipment. This is not only limited to the reaction and analytical equipment but also environmental sensors located in the laboratory. Each device has its corresponding input agent transmitting the data into the KG. The monitor agent is responsible for monitoring the status of the digital twin and assessing if further optimisation is required. If needed, it invokes the DoE agent to suggest new experiments and update the configurations of the digital twin. The actuation of such settings is the responsibility of the execution agent to reflect the changes made in the KG. This loop of self-optimisation continues until the monitor agent decides the optimal condition is reached. Importantly, with agents expressed in the OntoAgent format, this framework supports agent discovery service to enable agent-agnostic execution requests.

Compared to the platform-based approach, one distinguishing feature of the dKG approach is that everything is connected, scalable, unambiguous, distributed, multi-domain, interoperable, accessible, and most importantly evolving in time. As all the digital replicas of the hardware are expressed in the same way, new equipment can be immediately accessed by any existing software once it is instantiated in the KG. The same applies when adding new ML algorithms wrapped following OntoAgent specifications – standardised interactions with data and HPC services can be established in no time [248]. This enables the rapid integration of the most advanced algorithms and equipment. Due to the modularised nature, in contrast to heavily intertwined coding logic within a monolithic application, the duty of development of each component is separated, improving the maintainability of the entire system.

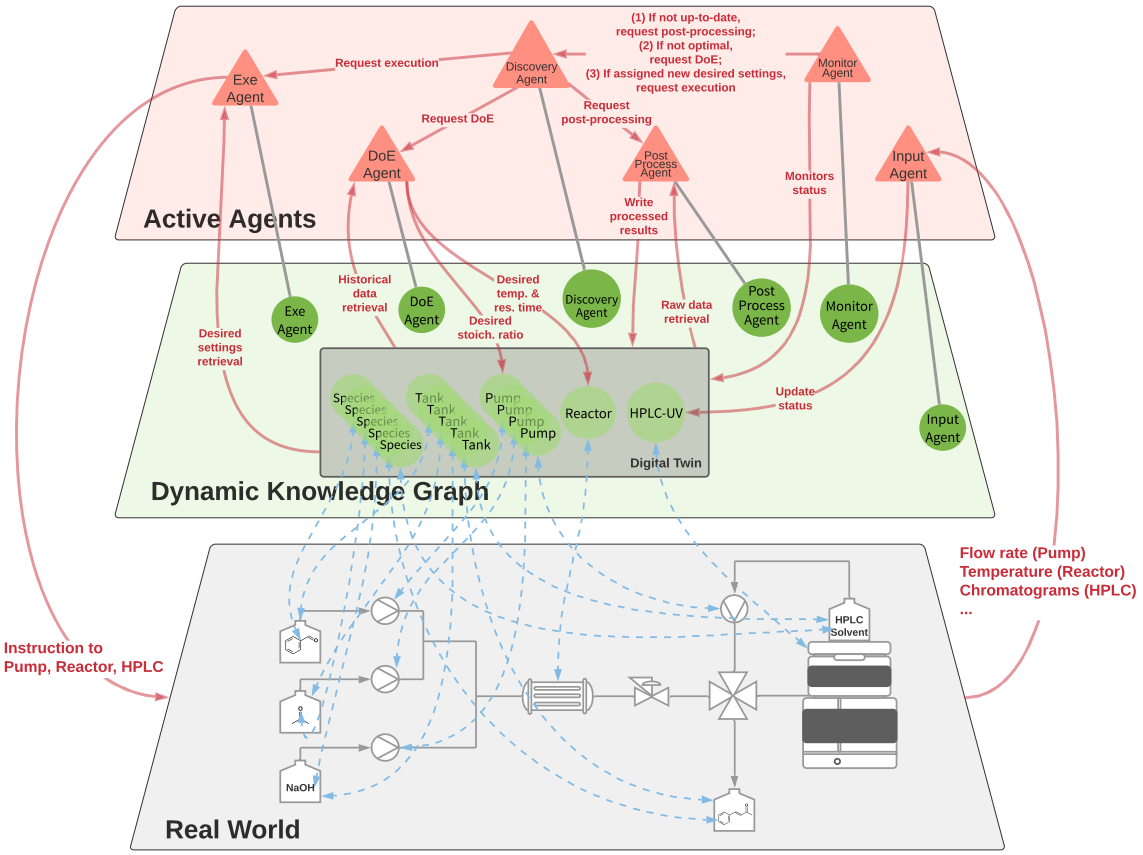


Fig. 3.3 The dKG approach towards automated closed-loop optimisation. The real world layer demonstrates the existing physical entities, adapting from the experimentation setup of Jeraal et al. [178]. The dynamic knowledge graph layer hosts all the data generated during the experimentation and a digital twin of the experimentation apparatus. This layer is dynamic as it reflects and influences the status of the real world in real-time. This synchronisation is enforced by the agents in the active agents layer which are instantiated from their ontological representation in the knowledge graph.

Another advantage of this approach is its future-proof nature, *e.g.*, its interoperability when integrating with other ontological initiatives in the community. At the species level, OntoSpecies acts like a register system that covers most of the chemical identifiers, making it possible to match with PubChemRDF or other molecular databases. In terms of chemical reactions, OntoKin is already able to describe the kinetic mechanisms of gas-phase chemistry. These concepts can be expanded to describe other chemistry domains of interest. A further opportunity lies in linking the reactions with concepts as defined in RXNO and MOP, embracing their full semantic capabilities. Similar expansion can be made with CHMO or AFO to describe the analytical data and method employed in the experimentation.

3.3.3 Towards a digital laboratory and beyond

In light of the envisioned prospects for the forthcoming era of a global collaborative network [300, 301, 172, 146, 108, 61, 382, 32, 342, 270, 311], we discuss the ways in which dKG can effectively tackle the four major challenges faced by contemporary SDLs as introduced earlier in this thesis.

Orchestration of heterogeneous resources As aforementioned, dKG leverages ontologies to abstract entities within the laboratory, thereby expressing their digital counterparts in the virtual realm. Moreover, the incorporation of software agents as APIs for the hardware strengthens the integration process. This approach effectively bridges the gap between software and hardware, facilitating seamless coordination between physical experiments and simulations and promoting better utilisation of the plethora of computational power in our efforts towards a sustainable future [128]. Furthermore, if the hardware components are further modularised with industrial communication standards, this layered abstraction holds the potential to unlock scalability among heterogeneous resources across SDLs. Such a harmonised and standardised framework promises to significantly enhance the efficiency and efficacy of research and development processes within diverse domains.

Data generation, integration, and utilisation The adoption of a unified representation of experimental resources in dKG paves the way for its effective storage of produced data. This, in turn, facilitates the integration of data from diverse devices, provided there is a consensual description of the experiment. Moreover, by converting literature data stored in open-source databases into the ontological format and merging it with newly generated data, a holistic framework for capturing and exchanging data is established. Additionally, the semantic-rich nature of the dKG incorporates prior knowledge into the data, thereby unlocking the potential to leverage informed ML applications [372].

FAIR workflow to facilitate reproducibility Similar to the vision of Roberts et al. [300, 301], dKG technology promotes information to be accessible to all stakeholders within a laboratory environment. For instance, active agents in the World Avatar share the same worldview. They communicate with pointers to the correct resources, *i.e.*, IRIs. This enables asynchronous communication to accommodate time-consuming activities. Moreover, the communication itself can be stored in the dKG and accessible to all agents – this ensures not only data but also workflows are transparent and FAIR, facilitating reproducible experimentation. By further introducing dependency between different concepts, both data and instructions to the instrument will act like a flow of information travelling in the dKG, analogous to an adaptive organism.

Open infrastructure for open science As previously discussed, different approaches towards SDLs coexist. Choices are to be made for groups upgrading from a common lab environment. Therefore, interoperability is key towards a network of connected SDLs. By design, the dKG approach utilises ontologies abstracted from the laboratory entities, it is possible to instantiate a new lab into dKG following the abstracted knowledge representations. Developing such ontologies is an iterative process and requires the consensus of the domain in developing and maintaining its life cycle. As demonstrated by the general semantic web community [157], and particular application experience in the chemical engineering community (OntoCAPE [242]), trial-and-error will be inevitable in this process. However, it is reasonable to be positive given the successful adoption of these technologies by giant IT companies [261]. In that regard, the World Avatar is an open project with all resources available on GitHub and welcomes contributions from the community.

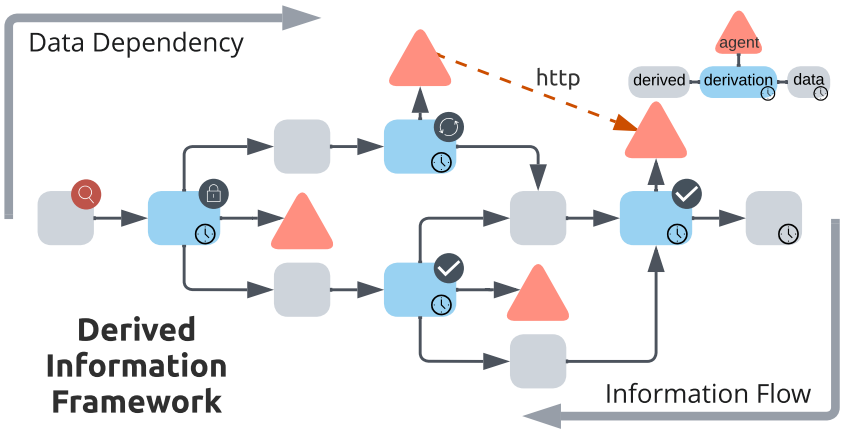
3.4 Chapter summary

In conclusion, we performed a thorough review of the data flow between the different functional components within state-of-the-art studies on SDLs. We found the common platform-based approach employs *ad hoc* data representations and subsequently different data transfer protocols. This results in scalability issues when integrating new hardware and software, and interoperability issues when collaborating among different platforms – better data representation and exchange are desired. We subsequently reviewed both semantic and non-semantic efforts in this regard and outlined the connections between initiatives. Besides the existence of a pattern to promote semantic representations of chemical knowledge, studies emerging to use agent-based approaches for standardised generation and consumption of data. With our past experience in closed-loop optimisation and KG development, we conjecture

that a dKG approach will bridge the existing gap resulting from the absence of standardised data representations and communication protocols. This would enable rapid integration of data and AI-based agents for chemical discovery and development, thereby advancing the realisation of a global collaborative research network.

Chapter 4

Automated provenance annotation and update



"I'll call "Society of Mind" this scheme in which each mind is made of many smaller processes. These we'll call agents. Each mental agent by itself can only do some simple thing that needs no mind or thought at all. Yet when we join these agents in societies–in certain very special ways–this leads to true intelligence."

– Marvin Minsky, *The Society of Mind (1986), p. 17*

The chapter draws from a paper published in *Future Generation Computer Systems* in collaboration with the Computational Modelling Group at the University of Cambridge, CMCL Innovations, and Cambridge CARES. Dr Lee and Dr Mosbach designed and developed the first version of the synchronous mode of the framework. Mr Hofmeister developed domain ontologies and agents used in the flood impact assessment use case. The asynchronous mode of the framework and the agent template that unifies a-/synchronous communications were developed by the author with feedback (code reviews) from Dr Mosbach, Dr Lee and Mr Hofmeister. The original draft was written by the author and all authors provided feedback.

In this chapter, a derived information framework is developed to semantically annotate how a piece of information can be obtained from others in a dKG. We encode this using the notion of a "derivation" and capture its metadata with a lightweight ontology. We provide an agent template to monitor derivations and to standardise agents performing this and related operations, making the KG truly dynamic. We implement both synchronous and asynchronous communication modes for agents interacting with the dKG. When occurring in conjunction, directed acyclic graphs of derivations can arise, with changing data propagating through the dKG by means of agents' actions. While the framework itself is domain-agnostic, we apply it to a case study in the smart cities domain and demonstrate that it is capable of handling sequential events across different timescales.

4.1 Introduction

Inspired by semantic web technology, KGs are gaining popularity both in enterprise applications [261] and research fields [157]. They are seen as a suitable approach to integrating diverse information sources and fostering common understanding among domain experts [139, 161]. Some renowned examples of static KGs are DBpedia [206] and Wikidata [276].

The dynamic aspect of KGs has recently been discussed [3], where they are envisaged as hubs for integrating complex systems of software agents, connecting different domains in specific use cases. This allows for what-if scenario analysis and automated decision-making that mimics human behaviour. This is a step towards the original vision of the Semantic Web, which is a fully annotated web of machine-readable data that can be processed autonomously by software agents [154, 21]. However, this also highlights the need for robust methods to manage the changes and interdependencies in the complex information network, especially in a data-rich world which is rife with misinformation.

A key enabling factor identified by the community is provenance [397], *i.e.*, where a piece of information originates from and how it came about. Provenance covers many aspects, including published literature, the wider internet, or data directly acquired through a measurement device. A number of provenance ontologies have been developed in the literature, for example, PAV [54], W3C Provenance Ontology (PROV-O) [204], Dublin Core Terms (DC Terms) [71], Bibliographic Ontology (BIBO) [70], and Open Provenance Model Ontology (OPMO) [244]. For a comprehensive review of developments in provenance data models, interested readers are referred to Sikos and Philp [331].

In dKGs, one can consider a specific sub-problem of provenance, namely when some pieces of information are directly calculated from, or derived from, other pieces of information by software agents, all of which are already stored in the KG. When multiple pieces of information depend on each other in a certain way, the resulting cascade of information progresses in time via a series of coordinated agent communications, where the calculated values of one process are inputs for other subsequent processes. To make the KG truly dynamic, the system should also enable automated propagation of perturbations in source input data. This introduces another point of view – caching. In addition to providing functions (agents) to calculate a result, that result is stored, *i.e.*, cached, in the KG. Many of the technical challenges are similar, such as being able to detect when the cache is out of date, and providing a function to update, refresh, or recalculate the cache.

Any approach to solving this problem can benefit from the lessons learned in other fields. In scientific computing, such a composition of computing tasks is referred to as *workflow* or *pipeline* [395, 74]. Workflows are typically expressed as Directed Acyclic Graphs (DAGs), where the nodes represent tasks and edges represent data flows [216]. Each task can only be started if all its precedent tasks are completed. There are many different Workflow Management Systems (WMSs) that can be used to orchestrate these tasks, ranging from traditional paradigms like Pegasus [73, 75] and Kepler [6] to modern approaches like Apache Airflow [356], Nextflow [82], and Lightning AI [97], to name a few. Some studies have focused on adaptive workflows, where changes in the input data may be incorporated into computation on-the-fly to provide real-time responses to dynamic events [220]. However, these systems often rely on heterogeneous and unstructured data models without semantic annotations [76], which can make it difficult to achieve interoperability across different systems, impeding the unification of the workflow community [65].

Another area which we may learn from is microservice architecture [85]. It is a Service-Oriented Architecture (SOA) that emphasises loose coupling and high cohesion. It involves having each service in the ecosystem be independently developed, deployed, and maintained by a small dedicated team. When an *event* occurs, which is a similar concept as an exe-

cution instance of one pre-defined workflow, microservices are composed and coordinated via message-passing. This architecture places no restrictions on developers regarding the technology used to implement each microservice as long as a unified interface is agreed upon. Nonetheless, it places significant demands on the network to ensure successful communication. In this paradigm, Distributed Application Runtime (Dapr) [67] uses a sidecar to simplify communication via direct or event-based publish/subscribe messaging, unifying both modes of communication on the same platform.

Taking notes from these advents, we may summarise and suggest a KG-native solution. By design, data in KGs can be uniquely identified via IRIs. All the active agents share the same worldview once granted access privileges. Analogous to the *message bus* in the microservice architecture, communication between agents operating on the KG can be delegated via serving the correct IRIs. This eliminates the necessity for large peer-to-peer data transfer. It can be further combined with the idea learned in the scientific workflow to model computation dependencies as DAGs in the dKG. By encoding agents' messages as provenance records, we remove the need for direct agent-to-agent communication and allow information to travel through the dKG. This decouples the system and allows for a distributed ecosystem of agents without the need for them to be aware of each other's existence.

The purpose of this chapter is to provide a proof-of-concept for a technology-agnostic implementation of a Derived Information Framework (DIF) for dKGs. DIF is a realisation of such a KG-native architecture. It uses a lightweight ontology to mark up provenance, an agent template to standardise agent operations, and an automated framework to propagate information changes in a dKG. The design aims to lower the entry barrier for researchers to model any real-world cascading events with minimum effort by providing a user-friendly template. The significance of this framework lies in its dual capability to not only track and document the calculation process but also to automatically re-execute the computation when accessing outdated information. In an era characterised by both complexity and an abundance of data, the framework enables automated integration of the rapid influx of new information, ensuring constant access to up-to-date insights into the subject of interest.

In particular, we demonstrate this through a flood impact assessment use case. The selection of the smart cities domain is motivated by its necessity for handling both fast and slow urban dynamics [239], making it an ideal context for showcasing the framework's capability to accommodate real-world events with varying frequencies. Flooding events, as well as other natural disasters, require fast decision-making and a holistic perspective to respond. Hence, having up-to-date information at all times is valuable. However, it is essential to underline that the framework's versatility extends beyond the smart cities domain. The application of this framework in SDLs will be demonstrated in the following chapters.

4.2 Methodology

This section provides an overview of the technical aspects of DIF. We begin by introducing the ontology created for annotating the provenance markup, then presenting the agent template that developers can use as a starting point when developing new agents. Lastly, we discuss a client library, which can be used to manage the derivation instances.

4.2.1 Derivation ontology

We refer to *derivation* as the record for a singular occurrence of the fact that some pieces of information are derived or calculated from some other pieces of information. The term *derivation subgraph* is used to describe the subgraph of all derivation-related markup when a collection of interdependent processes is represented. Since the information in a KG is captured in the format of Subject–Predicate–Object statements (triples), storing the markup to capture this fact can be considered as a way of attaching arbitrary metadata to triples [218]. For that, generic solutions have been developed or are currently in development, such as reification [145] and W3C's RDF-star [389], which at the time of writing is still at the draft stage. While some implementations exist, *e.g.*, Blazegraph's Reification Done Right (RDR) [24, 145] or more recently GraphDB [266], at present these are not sufficiently widely supported for implementing DIF in a technology-agnostic way, without tying ourselves to a particular product. Therefore, we choose to state the required metadata explicitly as triples and introduce OntoDerivation as a lightweight ontology to serve this purpose. We first discuss the relevant ontologies on provenance for workflows from which we draw inspiration. We then explain the TBox of OntoDerivation, followed by an example instantiation of the ABox. The connection between OntoDerivation and OntoAgent [401], an ontology used to define the capabilities of agents, is further demonstrated by how they may be used in conjunction to govern agent actions. Finally, we discuss the interoperability between OntoDerivation and PROV-O.

Provenance for workflows

As the *de facto* standard, PROV-O [204] adopts three base classes for provenance descriptions, *i.e.*, `prov:Entity` for the things subject to description, `prov:Activity` for the events occurring over a duration that lead to transformation of the entities, and `prov:Agent` for the things responsible for carrying out activities. PROV-O also provides qualified terms as elaborated information on binary relations between these base classes. For example, `prov:Association` qualifies the relationship `prov:wasAssociatedWith` between

`prov:Activity` and `prov:Agent` by pointing to an instance of `prov:Plan` using the relationship `prov:hadPlan`. This qualification indicates the steps of action undertaken by the agent to achieve its goals. However, `prov:Plan` is provided as a rather isolated concept, lacking further specifications on how it can be connected to other concepts that are relevant for execution. Hence, PROV-O appears more towards retrospective recordings of events rather than actively invoking agents to perform tasks.

P-Plan [118] partially bridges this gap by extending PROV-O with terms `p-plan:Step` and `p-plan:Variable` in scientific processes. This expansion helps to publish the methods and processes of scientific workflows as linked data. This work is later combined with Open Provenance Model (OPM) [245], leading to Open Provenance Model for Workflows (OPMW) [119] which supports the connection between a workflow template and its concrete executions. Notably, OPM was a legacy data model developed from the Provenance Challenge series [243] and was actually the reference for the creation of PROV-O [119]. The authors of OPMW acknowledged that one aspect that is not yet supported by OPMW is the automatic re-execution of the published workflows in heterogeneous computing environments [119]. In that regard, modern containerised solutions and cloud services can be a potential solution [49]. The ideas behind these works served as inspiration for our KG-native approach.

The decision not to directly reuse concepts from PROV-O was made to facilitate the implementation for the specific aspect of provenance (and their updates) that we are considering in this work – how a piece of information is calculated from other sources and when it occurs. Specifically, for the following reasons:

- The need to accommodate diverse temporal scales in the response time with regard to activities performed by the agents. This differentiation is essential for facilitating automated updates of derived information through software agents, specifically concerning synchronous and asynchronous processes as the latter requires the recording of job status. This relates to the TBox level.

- The requirement for a unified representation of timestamps in denoting the timeliness of both pure inputs and derivation instances, which will simplify the implementation of the updating algorithm in determining whether a derivation is outdated. PROV-O annotates these timestamps using different data properties with the range `xsd:dateTime`, leading to additional conversion for handling derivations instantiated from different timezones. This relates to the instantiation of provenance records.

- Instantiating every piece of information as `prov:Entity` makes it impossible to distinguish concepts in the agents' I/O signatures, let alone differentiate the capabilities of distinct agents in a complex derivation subgraph. One possible workaround is declaring

all concepts in the I/O as subclasses of `prov:Entity`, however, this introduces an additional level of abstraction to basically all concepts in the domain ontologies, which is unnecessary in our opinion. We discuss our solution in the design of TBox and elaborate on agents' I/O in discussions related to OntoAgent.

OntoDerivation TBox

Figure 4.1 depicts two types of derivation, *i.e.*, synchronous (`Derivation`) and asynchronous (`DerivationAsyn`) to accommodate situations that respond in different timescales when a request is received. The synchronous mode communicates via the endpoint exposed by the software agent. It is thus faster and hence intended for applications demanding real-time responses. The asynchronous mode communicates exclusively through the dKG. It has the advantage of recording each stage of the operation in the dKG, but it is slower and hence better suited to relatively expensive jobs.

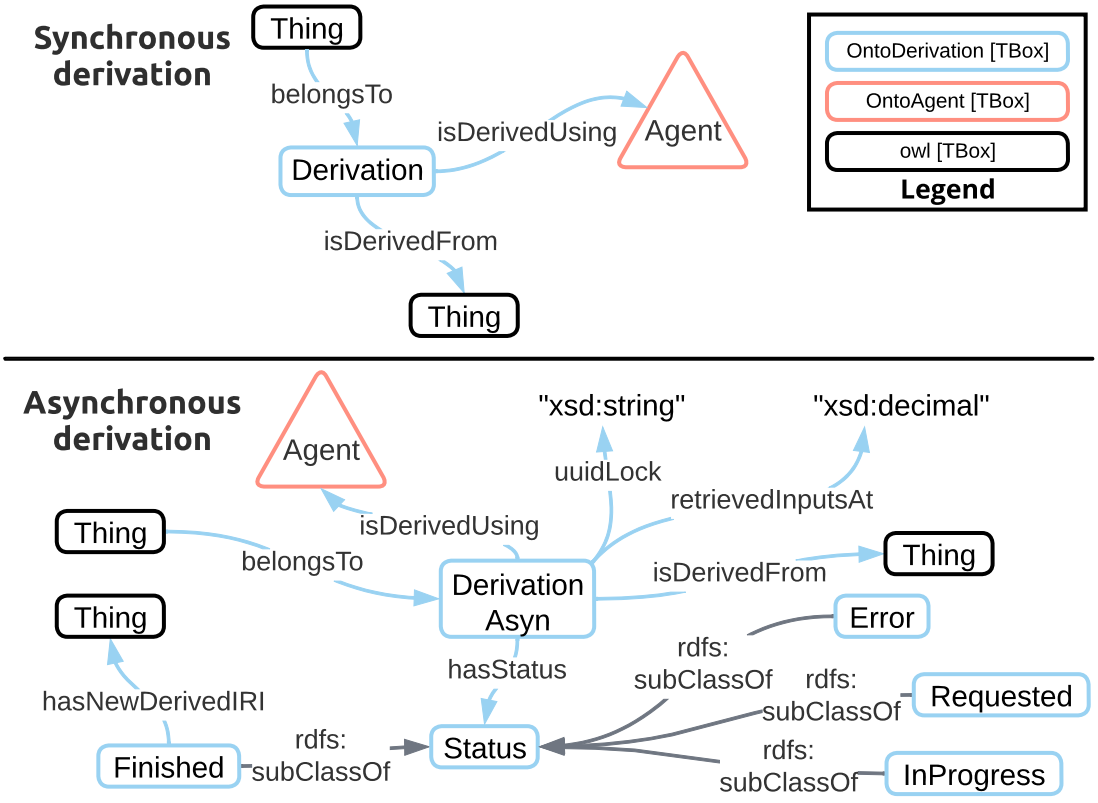


Fig. 4.1 Concepts and relationships of the OntoDerivation ontology. All classes and properties belong to the OntoDerivation namespace unless stated otherwise (for namespace definitions see Appendix B.1).

The information dependencies of a derivation are consistently marked regardless of the communication protocol, with the derived information (outputs) `belongsTo` the derivation, which itself `isDerivedFrom` some source information (inputs) and `isDerivedUsing` an agent defined in OntoAgent [401]. It is worth noting that there is no limit on the number of inputs or outputs for a derivation, but one output entity cannot `belongsTo` more than one derivation instance. Both source and derived information are abstracted using `owl:Thing`, so it is also possible for an input of a derivation to be part of another derivation instance, meaning that the input is a piece of derived information itself and `belongsTo` another derivation.

To support asynchronous operation, the concept `Status` is introduced to mark the state of an asynchronous derivation with the available options of `Requested`, `InProgress`, `Finished` and `Error`. The data property `retrievedInputsAt` records the timestamp when the inputs for the derivation were read in order to start the computation, which will later be used to update the timestamp for the derivation instance. The data property `uuidLock` uniquely identifies the agent thread that is processing the derivation and prevents any amendment from other threads that do not hold the correct key. This ensures thread-safe operations when multiple threads are employed to boost the throughput of derivation processing. A specific object property called `hasNewDerivedIRI` is used at the `Finished` status to temporarily link any newly derived information. These output entities will eventually be reconnected to the derivation instance after the agents clean up the status. Like the final outputs, there is no limit on the number of new entities that can be connected through `hasNewDerivedIRI` for each derivation instance.

HermiT reasoner [124] and OntoDebug plugin [314] for the Protégé ontology editor [254] were used for an evaluation. The OntoDerivation implements 8 classes, 5 object properties and 2 data properties. The HermiT reasoner is able to classify the OntoDerivation ontology. In the debugging mode, OntoDebug detects the ontology as both *coherent* and *consistent*. Protégé does not show any errors related to the illegal declaration of entities or reuse of entities. A description logic representation of OntoDerivation ontology is provided in Appendix B.2. The OntoDerivation TBox is version-controlled on GitHub[1].

OntoDerivation ABox

Figure 4.2 exemplifies an instantiated derivation instance and its simplified representation. Upon initialisation, each derivation is annotated with a timestamp following the W3C standard [64]. The Unix timestamp is chosen over `xsd:dateTime` to enable direct numerical comparison. In the rest of this thesis, simple integers will be used instead of the actual (Unix)

[1]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontoderivation/OntoDerivation.owl

timestamp for readability. Over the lifecycle of a derivation, this timestamp is used to assess whether it is out-of-date. In this example, as the timestamp of derivation is smaller than that of its source information, *i.e.*, $1 < 36$, we conclude that this derivation is outdated and that, hence, an update of the derived information is required.

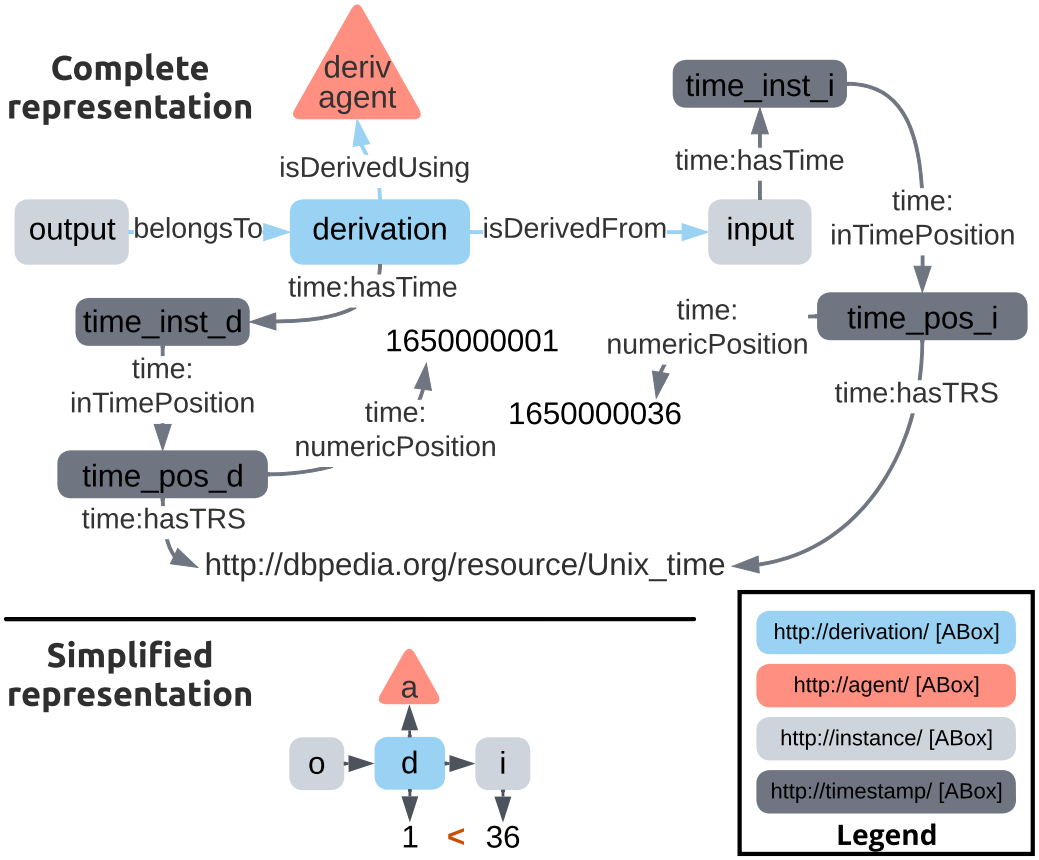


Fig. 4.2 An example derivation instance fully annotated with metadata and its simplified representation, which will be used throughout the rest of this thesis. All properties belong to the OntoDerivation namespace unless stated otherwise (for namespace definitions see Appendix B.1).

It should be noted, however, that though the outputs of a derivation can be input to others, they are not directly associated with a timestamp (but are indirect via the derivations to which they belong). For example, the *output* instance in Fig. 4.2 is not directly associated with a timestamp and so are all other instances that `belongsTo` a derivation. This design choice is made to reduce the redundant information in the dKG as the timeliness of the derived information is already reflected by the timestamp of the derivation instance it `belongsTo`. As a result, when a derivation subgraph is composed of several connected derivations, direct timestamp comparison becomes infeasible for those derivations whose inputs are outputs

of another derivation instance. It is therefore necessary to employ additional criteria for determining whether a derivation is out-of-date. More information on its implementation is provided in section 4.2.3.

Figure 4.3 illustrates three levels of generality in a derivation subgraph, from basic linear *chain* to non-linear *polytree* to generic *directed acyclic graph*. Unlike in scientific workflows, where the arrows often point in the same direction as the data flow, the markup in dKG denotes the data dependencies and points to the source of the information. The changes in the source information travel in the opposite direction within dKG, as illustrated by "information flow". In this context, we define *upstream* and *downstream* to refer to the relative location of a derivation instance within that flow. A key feature of the design is that only relevant downstream information is updated when accessed, which will be discussed in more detail in section 4.2.3.

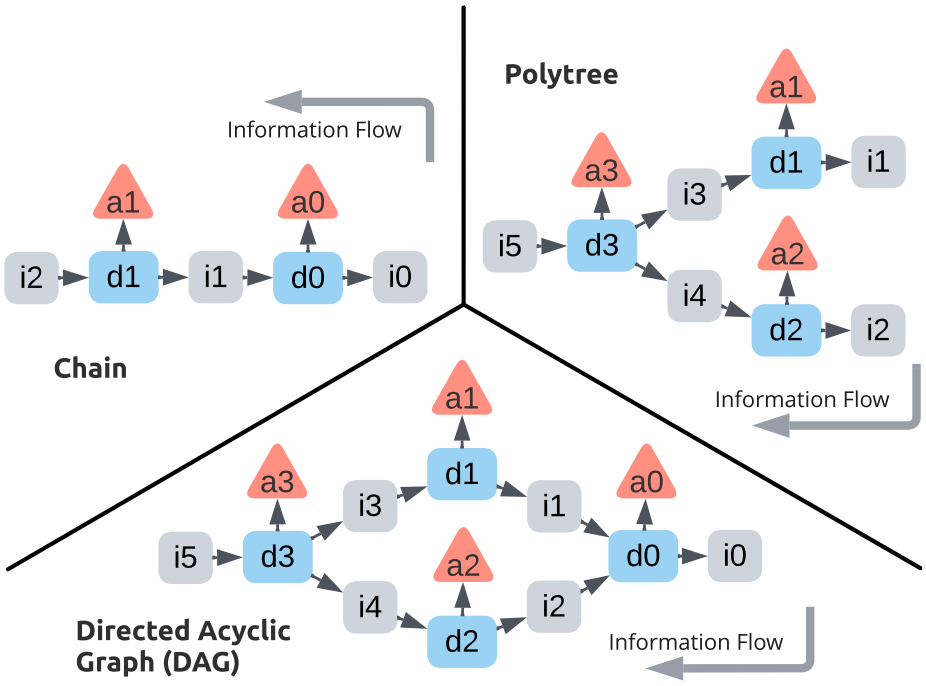


Fig. 4.3 Derivation subgraph structures as DAGs of varying generality. The arrows between instances indicate the markup for data dependencies. The "information" flows in the opposite direction, *i.e.*, from right to left.

We emphasise that the primary intent of DIF is to represent logical dependencies as such, as a historical record, rather than consider them as *steps* in a workflow or algorithm. This implies that, in contrast to workflows [196], cyclic dependencies are not permitted in DIF, as they would amount to logical contradictions. Nonetheless, the framework can of course be

used to record the dependencies of pieces of information that were obtained from algorithms containing loops and other circular constructs.

OntoDerivation ontology is designed in a way that is easy to use and easy to query. Tools are provided to automatically generate derivation markup for all three types, and validate the generated derivation subgraph. An example SPARQL query to retrieve all derivations in a given dKG is provided below:

```
PREFIX OntoDerivation: <https://www.theworldavatar.com/kg/ontoderivation/>
PREFIX time: <http://www.w3.org/2006/time#>

SELECT ?derivation ?devTime ?inputTime ?status ?status_type
WHERE {
  VALUES ?derivationType {
    OntoDerivation:DerivationAsyn
    OntoDerivation:Derivation
    OntoDerivation:DerivationWithTimeSeries
  }
  ?derivation a ?derivationType ;
  time:hasTime/time:inTimePosition/time:numericPosition ?devTime .
  OPTIONAL {
    ?derivation OntoDerivation:isDerivedFrom ?upstream .
    ?upstream time:hasTime/time:inTimePosition/time:numericPosition ?inputTime .
  }
  OPTIONAL {
    ?derivation OntoDerivation:hasStatus ?status .
    ?status a ?status_type .
  }
}
```

Connection with OntoAgent

In the World Avatar, OntoAgent [401] is used to mark up the I/O signature of agents to facilitate agent discovery. This markup points to concepts in domain ontologies and indicates the agent's capabilities by identifying the concepts required/produced by the agents. By contrast, OntoDerivation focuses on the instance level, *i.e.*, actual data digested and produced by the agents corresponding to each occurrence of computation, revealing the opportunity for employing both ontologies to regulate agent operations.

Figure 4.4 provides an example of instantiation using both OntoDerivation and OntoAgent, where inputs and derived outputs of the derivation instance are both instantiated

from the I/O signature defined in the OntoAgent instance. Specifically, the relationship `OntoAgent:hasType` can indicate that an agent's message encompasses various concepts from domain ontologies. For a given derivation instance, the inputs can be classified into key-value pairs, with the IRI of each concept as the key and the list of instances as the value. For example, the value of the pair with the key *DomainOntology_1:ExampleConcept_1* will be *input_1_1* and *input_1_2*. This design connects the conceptual capability of the agents with the concrete tasks that they are assigned to execute. Following this practice, the development of agents can be focused on the concept level, simplifying the implementation.

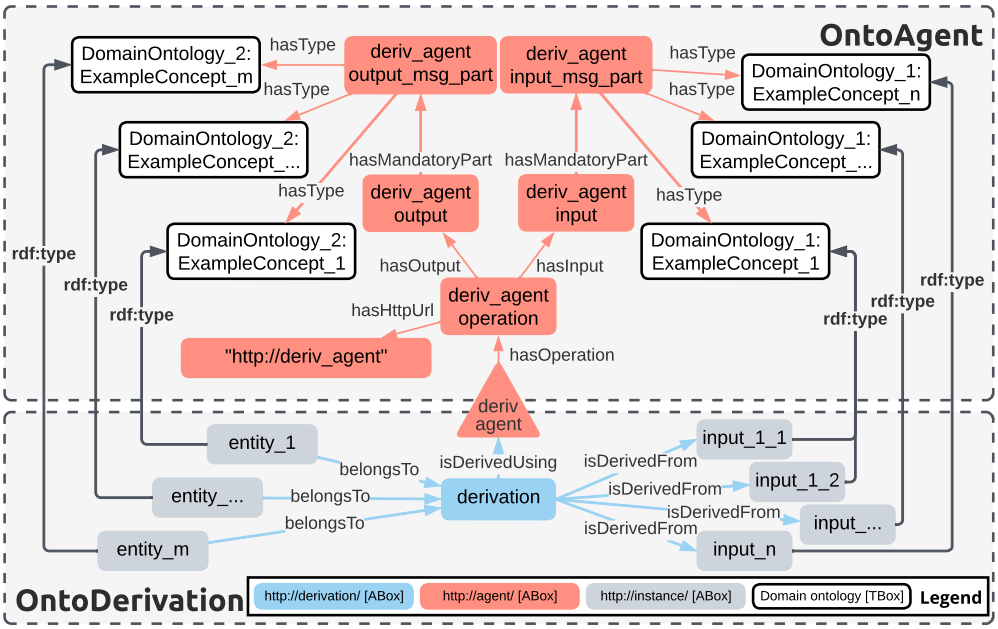


Fig. 4.4 The instantiation that connects OntoDerivation and OntoAgent. The linkage between derivation instances and agent instances is used to regulate agent operations. All object properties belong to the OntoDerivation namespace unless stated otherwise (for namespace definitions see Appendix B.1).

Interoperability with PROV-O

As aforementioned, PROV-O is a generic ontology for representing provenance records in various domains. Figure 4.5 illustrates a practical implementation to enable interoperability between OntoDerivation and PROV-O. The derivation graph of a given derivation and all its upstream derivations can be programmatically translated into corresponding PROV-O terms using the following SPARQL query. The resulting triples can be exported as linked data for publication and may be queried and processed using provenance tools[2].

[2]https://www.software.ac.uk/who-do-we-work/provenance-tool-suite

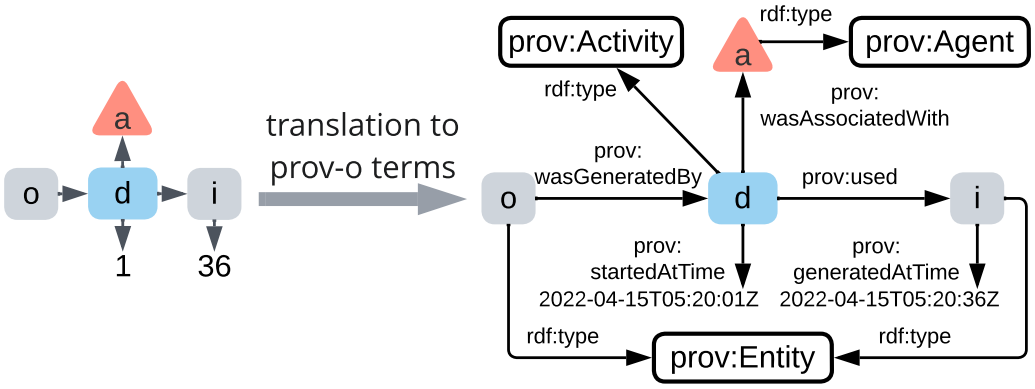


Fig. 4.5 An example translation of a derivation instance from OntoDerivation to PROV-O terms. The Unix timestamps are converted to `xsd:dateTime`.

```
PREFIX OntoDerivation: <https://www.theworldavatar.com/kg/ontoderivation/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX time: <http://www.w3.org/2006/time#>
PREFIX prov: <http://www.w3.org/ns/prov#>
CONSTRUCT {
  ?agent a prov:Agent . ?derivation a prov:Activity .
  ?input a prov:Entity . ?output a prov:Entity .
  ?derivation prov:wasAssociatedWith ?agent . ?derivation prov:used ?input .
  ?output prov:wasGeneratedBy ?derivation .
  ?derivation prov:startedAtTime ?devDateTime .
  ?input prov:generatedAtTime ?inputDateTime .
}
WHERE {
  <derivationIRI> (OntoDerivation:isDerivedFrom/OntoDerivation:belongsTo)* ?
      derivation .
  ?derivation OntoDerivation:isDerivedUsing ?agent .
  ?derivation time:hasTime/time:inTimePosition/time:numericPosition ?devTime .
  BIND("1970-01-01T00:00:00Z"^^xsd:dateTime + STRDT(CONCAT("PT", STR(?devTime), "
      S"), xsd:duration) AS ?devDateTime)
  ?derivation OntoDerivation:isDerivedFrom ?input .
  ?output OntoDerivation:belongsTo ?derivation .
  OPTIONAL {
    ?input time:hasTime/time:inTimePosition/time:numericPosition ?inputTime .
    BIND("1970-01-01T00:00:00Z"^^xsd:dateTime + STRDT(CONCAT("PT", STR(?inputTime
        ), "S"), xsd:duration) AS ?inputDateTime)
  }
}
```

One aspect of interoperability that remains unexplored is the capability to query and update provenance records in native PROV-O expressions. This pertains to the absence of status information in the exported provenance due to the lack of inherent support for status in PROV-O. The capability to possess this falls however beyond the scope of this work.

4.2.2 Derivation agent

The use of ontological markup to record each step in the process of updating derived information largely restricts the communication an agent needs to perform to the dKG itself, rather than with other agents. We provide an agent template to support this in both synchronous and asynchronous modes. The template makes use of data container classes to host agents' inputs and outputs. These classes are key-value pairs that may be mapped using the following SPARQL query:

```
PREFIX OntoDerivation: <https://www.theworldavatar.com/kg/ontoderivation/>
PREFIX OntoAgent: <http://www.theworldavatar.com/ontology/ontoagent/MSM.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT ?input ?type
WHERE {
  <agentIRI> OntoAgent:hasOperation/OntoAgent:hasInput/
    OntoAgent:hasMandatoryPart/OntoAgent:hasType ?type .
  <derivationIRI> OntoDerivation:isDerivedFrom ?input .
  ?input a*/rdfs:subClassOf* ?type .
}
```

Developers are supplied with utility functions to access/validate the mapped inputs and to construct outputs. The agent logic that computes outputs from the inputs is the only code required from the developer. We made the template available in both Java and Python to increase accessibility and depict its Unified Modelling Language (UML) activity diagram in Fig. 4.6. The essential design elements are elaborated on below.

Synchronous communication mode

The synchronous communication mode is realised through direct agent requests/responses. Upon receiving a request for a normal `Derivation`, the agent serialises the request content to an instance of the container class. The time instant is recorded immediately before passing the inputs to be processed by the developers' code. This instant is considered as the timestamp when inputs were read. Once the outputs are constructed, an update operation

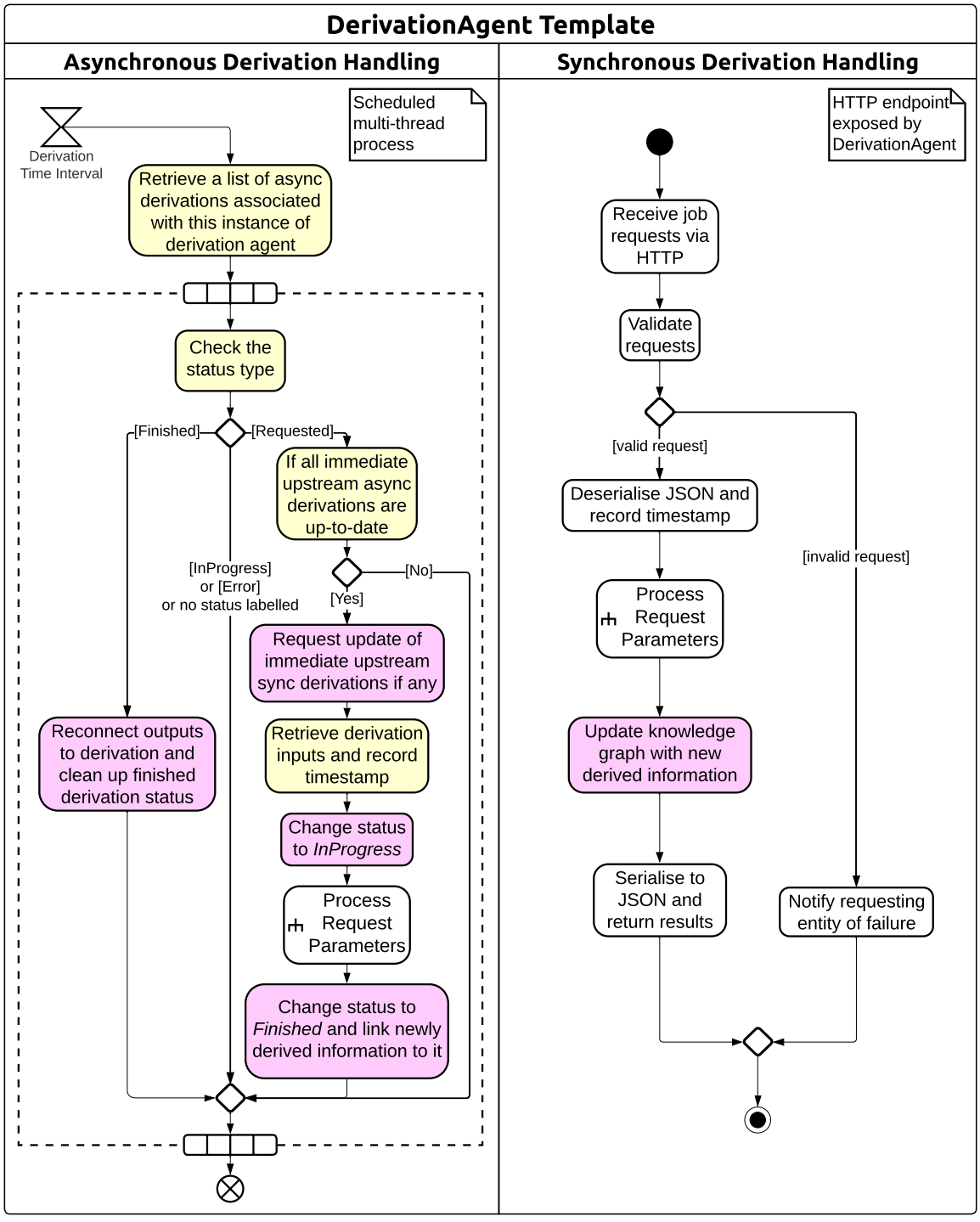


Fig. 4.6 UML activity diagram of the derivation agent template supporting both synchronous and asynchronous derivations. Developers need only supply the activity node `ProcessRequestParameters` for specific agent capabilities. The yellow- and magenta-shaded actions represent dKG data retrieval and population operations respectively.

will be formulated and executed by the agent to update the dKG. If there is no error, the derivation update is considered successful and a response will be returned that includes the produced derived information and the recorded timestamp.

Asynchronous communication mode

In the asynchronous communication mode, agents monitor the status of derivations in the dKG and perform any requested tasks. When an agent detects a derivation with the status `Requested`, it first checks if all the data dependencies for that derivation are satisfied.

If the requirements are met, the agent retrieves the inputs from the dKG, records the current timestamp, and changes the status of the derivation to `InProgress`. The inputs are then passed to the method provided by the developer and transformed into outputs. At this point, the status of the derivation will be changed to `Finished` and the newly derived outputs are temporarily connected to it. This status refers to a distinction made between a task that has been completed but still requires post-processing or cleaning-up, and a task that is complete in the sense that it requires no further action. It is used to prevent multiple agents from trying to perform the same cleaning-up tasks simultaneously and is removed when the derivation subgraph is tidied up during the next scheduled monitoring period. The cleaning-up process includes deleting the old instances, connecting the new instances with the original derivation and any downstream derivations that exist, removing all the status information, and finally updating the timestamp of the derivation to keep the derivation subgraph current. The monitoring is performed at a scheduled time interval and its frequency can be user-defined. If an error occurs during any operation, the agent changes the status of the derivation to `Error` and records the exception trace.

Concurrency and multi-threading

Being a decentralised system by design, the World Avatar contains many agents operating on the dKG simultaneously. In situations where multiple agents request updates to the same derivation, the corresponding agent must handle concurrent requests correctly and efficiently. For example, such a situation arises when instance *i1* and *i2* illustrated in the generic form of DAG in Fig. 4.3 are accessed at the same time. In synchronous communication mode, the current implementation ensures that no duplicated information is added to the dKG by using the following SPARQL update. In asynchronous communication mode, the agent uses the data property `uuidLock` to identify and lock the thread currently handling a derivation, avoiding duplicated execution. These measures ensure that concurrency is handled correctly without sacrificing the high throughput of multithreading for distributed agent deployment.

```
PREFIX OntoDerivation: <https://www.theworldavatar.com/kg/ontoderivation/>
PREFIX time: <http://www.w3.org/2006/time#>

DELETE {
  ?e ?p1 ?o . ?s ?p2 ?e .
  ?d OntoDerivation:hasStatus ?status . ?status a ?statusType .
  ?timeIRI time:numericPosition ?dTs .
}
INSERT {
  <newInstance1> a <rdfTypeOfNewInstance1> .
  <newInstance2> a <rdfTypeOfNewInstance2> .
  <newInstance3> a <rdfTypeOfNewInstance3> .
  <newInstance3> OntoDerivation:belongsTo <derivationIRI> .
  <newInstance1> OntoDerivation:belongsTo <derivationIRI> .
  <downstreamDerivation1> OntoDerivation:isDerivedFrom <newInstance1> .
  <downstreamDerivation2> OntoDerivation:isDerivedFrom <newInstance1> .
  <newInstance2> OntoDerivation:belongsTo <derivationIRI> .
  <downstreamDerivation3> OntoDerivation:isDerivedFrom <newInstance2> .
  ?timeIRI time:numericPosition 1659981343 .
}
WHERE {
  {
    SELECT ?d ?timeIRI ?dTs ?status ?statusType ?e ?p1 ?o ?s ?p2
    WHERE {
      {
        VALUES ?d { <derivationIRI> }
        ?d time:hasTime/time:inTimePosition ?timeIRI .
        ?timeIRI time:numericPosition ?dTs .
        ?d OntoDerivation:isDerivedFrom/OntoDerivation:belongsTo? ?ups .
        ?ups time:hasTime/time:inTimePosition/time:numericPosition ?upsTs .
        FILTER (?dTs < ?upsTs)
      }
      {
        ?e OntoDerivation:belongsTo ?d .
        ?e ?p1 ?o . OPTIONAL { ?s ?p2 ?e . }
      }
      OPTIONAL {
        ?d OntoDerivation:hasStatus ?status . ?status a ?statusType .
      }
    }
  }
}
```

4.2.3 Derivation client

Having established the ontology to capture the derivation process and the agent template to perform the derivation update, we introduce here the derivation client capable of managing the derivation subgraphs. This involves determining if a derivation is out-of-date and, if so, requesting an update. This section discusses three cases in which updates to the derivation subgraph are handled using different communication modes, namely: purely synchronous, purely asynchronous, and mixed-type. For each case, we first present the general algorithm and then provide examples to describe the intended outcome.

Purely synchronous update

Determining the timeliness of each derivation and performing the necessary updates in a derivation subgraph is a recursive process. Given any derivation instance where the accessed information is derived from, the framework treats it as root, traverses upstream all the way to the derivation that is derived from source information, *i.e.*, all inputs of whom are not derived from anything, and finally updates derivations backwards. This may be described as a Depth-First Search (DFS) algorithm, presented in Algorithm 1.

Figure 4.7 illustrates a notable detail of Algorithm 1 that uses the DAG G to track the traversing of the graph. To do this, the algorithm adds derivation instances as vertices and connections between them as directed edges to G. Depending on the root derivation, the DFS algorithm can result in two versions of G, leaving out parallel branches that do not require updating. The update is carried out only for the derivations in the resulting graph during the DFS algorithm's backtracking. It should also be noted that, the computation only proceeds when both node and edge are previously unseen. For example, regardless of which branch is traversed first (derivation *d1* or *d2*) in the upper resulting graph, derivation *d0* is only visited the first time when branching. This design ensures the relevant upstream information are only visited once to avoid duplication of work.

Figure 4.8 illustrates the simplest form of derivation update, *i.e.*, updating one synchronous derivation. For demonstration, we take the simplified representation of derivation expressed in Fig. 4.2 at timestamp 36 as a starting point. As aforementioned, this derivation is deemed outdated when comparing its timestamp with that of its inputs. Assuming the output information is accessed at 60, the framework fires an update request to the agent associated with the derivation instance. Upon receiving the update request, the agent starts a calculation immediately and updates the output entity in the dKG with the newly derived information. The derivation instance is thus up-to-date and the results are returned.

Algorithm 1: Update synchronous derivations

Input: IRI of the root derivation instance
Result: The root derivation and all its upstream derivations are updated if deemed outdated

Create an empty directed acyclic graph G;
Cache root derivation d and all its upstream derivations recursively in G;
`updateSyncDerivation(`d, G`);`

Function `updateSyncDerivation(`d, G`)`**:**

- $U \leftarrow d.upstreams()$; `/* get immediate upstream derivations */`
- **if** *vertex* $d \notin G.vertices$ **then**
  - Add d as vertex in G;
- **end**
- **for** $u \in U$ **do**
  - **if** *vertex* $u \notin G.vertices$ **then**
    - $visited_u \leftarrow false$;
    - Add u as vertex in G;
  - **else**
    - $visited_u \leftarrow true$;
  - **end**
  - **if** *edge* $(d,u) \notin G.edges$ **then**
    - $traversed_{d,u} \leftarrow false$;
    - Add (d,u) as edge in G ; `/* will throw error if circular dependency detected */`
  - **else**
    - $traversed_{d,u} \leftarrow true$;
  - **end**
  - **if** $visited_u == false$ **and** $traversed_{d,u} == false$ **then**
    - `updateSyncDerivation(`u, G`);`
  - **end**
- **end**
- Fire request to update d if it is deemed outdated;
- Update outputs of d in cache;

To expand this example to a slightly more complex situation, we consider accessing information *i2* in a linear chain of two synchronous derivations, as represented in Fig. 4.9. In this example, the input information of derivation *d1* belongs to derivation *d0*, therefore, the timeliness of *d1* is determined by comparing it with the derivation *d0*. However, just comparing the timestamps of these two will lead to an incorrect conclusion that *d1* is up-to-date. On the contrary, *d1* should be considered as outdated as it depends on an outdated derivation. Therefore, in the situation of assessing the timeliness of derivations that depend

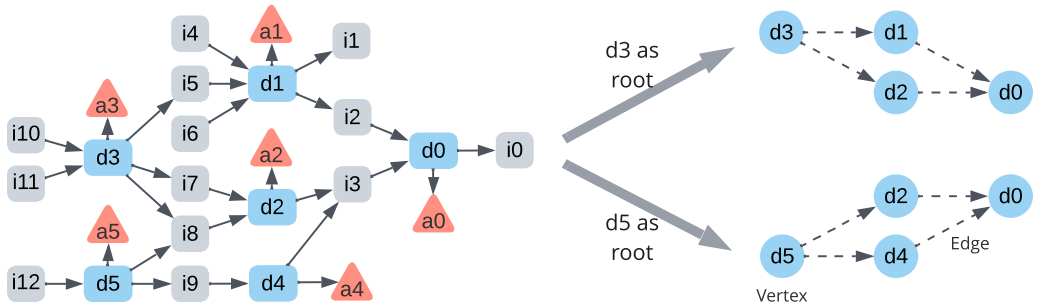


Fig. 4.7 DAGs are used in memory to assist the backtracking of the DFS algorithm when updating the derivation subgraph. The derivation instances are treated as vertices and their connections as edges. Depending on the root derivation chosen, different DAGs can be generated.

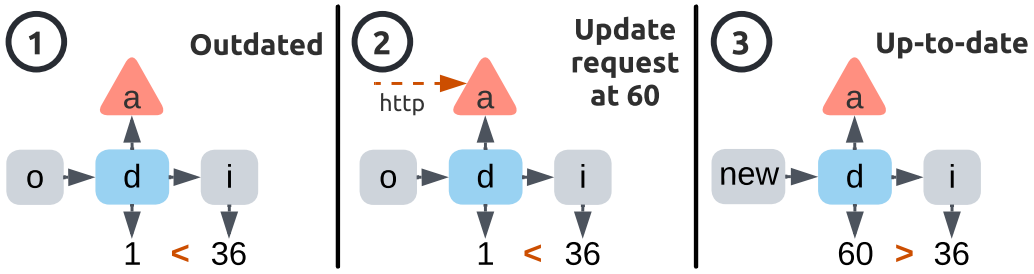


Fig. 4.8 The process of updating a single synchronous derivation.

on derived information, the framework determines the timeliness of the upstream derivation (*d0*) first before performing any action to the downstream derivation (*d1*). For the presented example, assuming that information *i2* is accessed at time 80, the framework updates *d0* first, then *d1* immediately afterwards. With the new outputs connected in the dKG, both derivations will be seen as up-to-date.

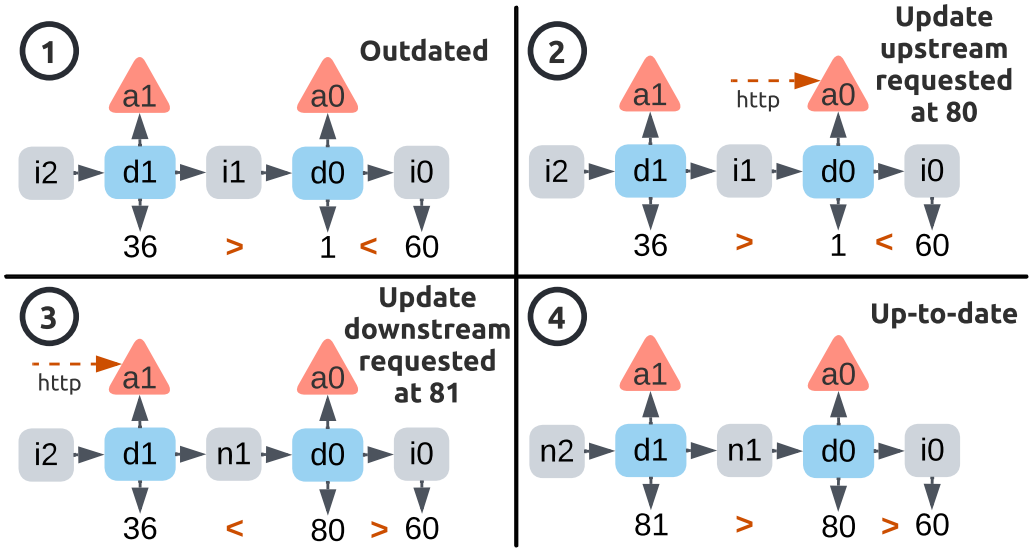


Fig. 4.9 The process of updating a derivation DAG consisting of only synchronous derivations.

Purely asynchronous update

In the scientific computing domain for example, it is common to have situations where lengthy calculations from source input data are requested, requiring minutes or hours of wall time before the outputs are available. The asynchronous update suitable for such situations is discussed in this section, as described in Algorithm 2. It is similar to Algorithm 1, except that the algorithm for asynchronous derivations does not cache the derivation subgraph. Rather, the immediate upstream derivations are queried on-the-fly and hence their timeliness is determined purely based on real-time queries of the dKG. The purpose of this design is to account for the fact that, due to the relatively long time scales involved, any information in the subgraph of asynchronous derivations can change while the process of carrying out an update is taking place.

As depicted by Fig. 4.10, we start from the point in time when the derivation is just instantiated, *i.e.*, the asynchronous derivation is initialised with the status as `Requested` and a timestamp of 0, with no output computed. The actual update of an asynchronous derivation will be dealt with by the derivation agent and is not concurrent with the request for that update. As the agent periodically checks the status of derivations that are derived using itself, the requested derivation will be turned into `InProgress` at the next trigger and the timestamp when the inputs were read will be recorded. The successful completion of the job will be reflected in its status `Finished`. The agent will then connect the generated output to the derivation instance, update the timestamp and lastly remove the status altogether. The update process is similar to that of the initial computation. The only difference is that the

Algorithm 2: Update asynchronous derivations

Input: IRI of the root derivation instance
Result: The root derivation and all its upstream derivations are requested for update if deemed outdated
Create an empty directed acyclic graph G;
Query derivation instance d from the KG using the given rootDerivationIRI;
`updateAsyncDerivation(`d, G`);`

Function `updateAsyncDerivation(`d, G`):`
  Query the list U of all immediate upstream derivations of d;
  if *vertex* $d \notin G.vertices$ **then**
    Add d as vertex in G;
  end
  for $u \in U$ **do**
    if *vertex* $u \notin G.vertices$ **then**
      $visited_u \leftarrow false$;
      Add u as vertex in G;
    else
      $visited_u \leftarrow true$;
    end
    if *edge* $(d,u) \notin G.edges$ **then**
      $traversed_{d,u} \leftarrow false$;
      Add (d,u) as edge in G ; `/* will throw error if circular dependency detected */`
    else
      $traversed_{d,u} \leftarrow true$;
    end
    if $visited_u == false$ **and** $traversed_{d,u} == false$ **then**
      `updateAsyncDerivation(`u, G`);`
    end
  end
  Mark d as Requested if it is deemed outdated;

agent will perform instance matching when reconnecting the newly derived information to existing downstream derivations in the derivation subgraph.

The same example can be expanded to a linear chain of two asynchronous derivations instantiated with only one piece of input data, as illustrated in Fig. 4.11. In this case, as none of the outputs are computed, the downstream derivation *d1* is directly marked as `isDerivedFrom` its upstream derivation *d0*. The agent responsible for *d0* will operate in the same way as aforementioned, the only difference being that the agent will reconnect the new derived instance *i1* as input to derivation *d1* and remove the direct connection between

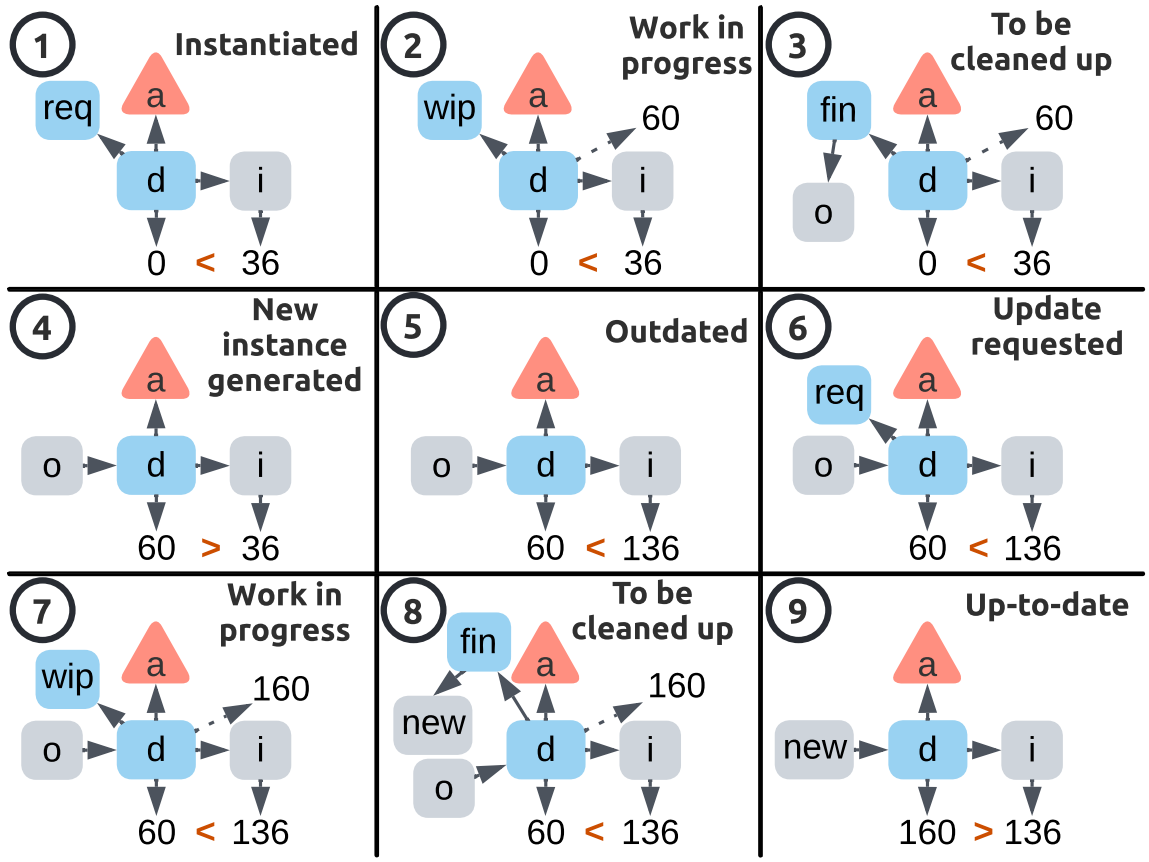


Fig. 4.10 The process of updating a single asynchronous derivation. The integers attached to the derivation instances via the dashed arrows denote the timestamps recorded by the data property `retrievedInputsAt`.

the two derivations. Similar to scientific workflow, the agent that manages the downstream derivation *d1* can begin its execution only after its predecessors have finished successfully. Therefore, block 1 to 7 of Fig. 4.11 showcases one desired usage of DIF to automatically complete a predefined workflow given input data. Once all derived instances are computed, we illustrate the steps for updating the derivations in block 8 to 15, where the source input data is updated. Upon request, the algorithm traverses the derivation chain and determines the timeliness of the derivations. Unlike synchronous update, the algorithm only marks derivations as `Requested`, leaving the actual update to individual agents in the same way as aforementioned.

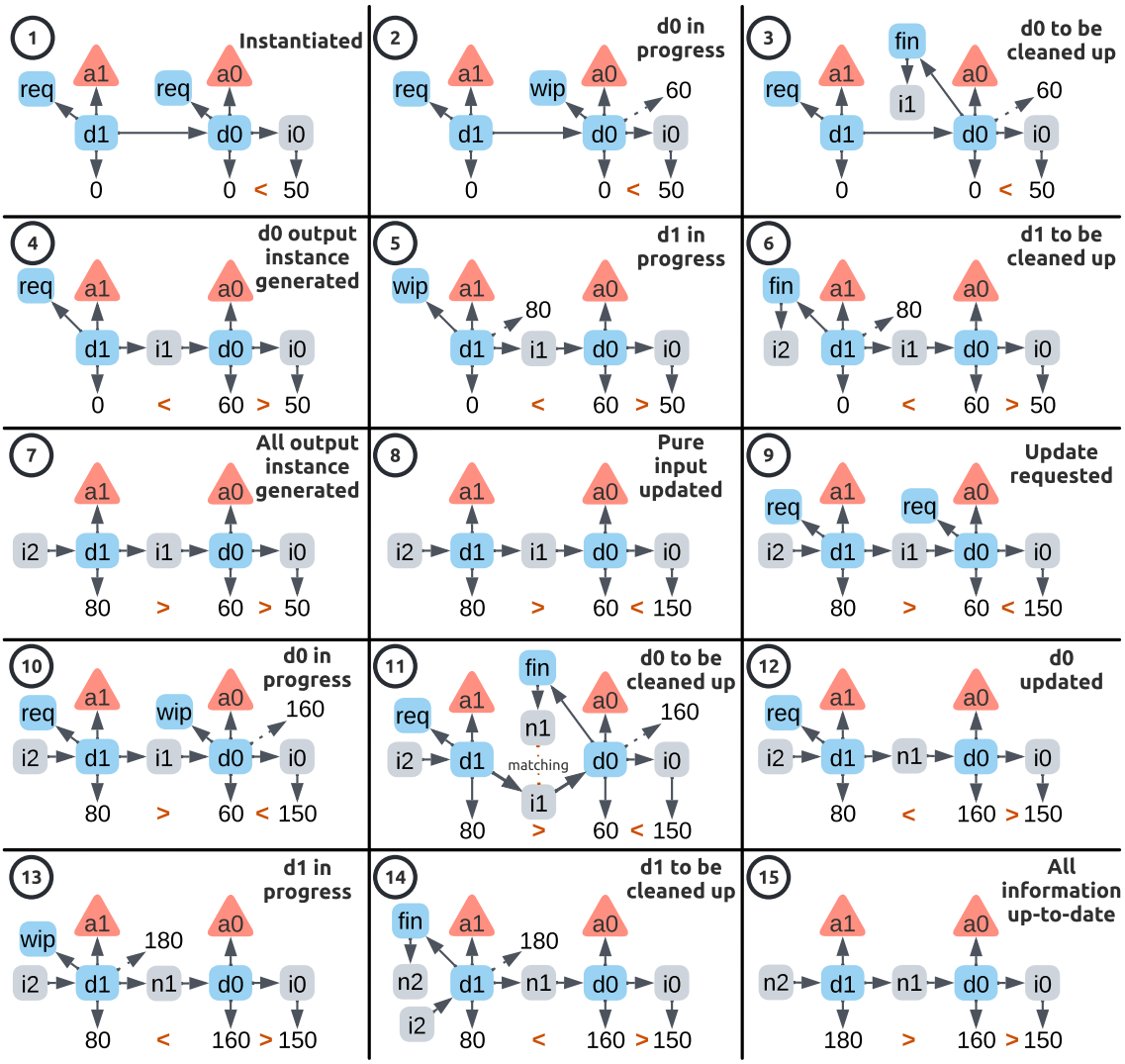


Fig. 4.11 The process of updating a derivation DAG consisting of only asynchronous derivations. The integers attached to the derivation instances via the dashed arrows denote the timestamps recorded by the data property `retrievedInputsAt`.

Mixed-type update

The final example we provide is a derivation subgraph consisting of mixed-type derivations. Specifically, asynchronous derivations depend on synchronous derivations, *i.e.*, the lengthy calculation relies on results from fast computations. It is worth noting that the other way would lead to a purely asynchronous scenario. If a synchronous derivation is dependent on an asynchronous one, the synchronous derivation effectively becomes asynchronous due to the need to wait a long time for the response. For that reason, the case of asynchronous

derivations depending on synchronous ones is the only mixed case that needs to be considered, without loss of generality.

As previously shown, we need to determine if a derivation is out-of-date and perform/request an update. These two parts are combined and executed in a recursive algorithm for derivation subgraphs that only consist of synchronous derivations. However, to save computation resources for updating mixed-type derivation subgraphs, we would like to design the framework to work in a way that the update of upstream synchronous derivations is only computed when the agent updates the asynchronous derivation. Therefore, when the algorithm recursively determines the timeliness of the asynchronous derivations, markup is needed in the dKG to indicate that the downstream asynchronous derivation is in fact out-of-date and, hence, should be marked as `Requested`.

A unified method is provided as a wrapper function of the two algorithms previously introduced. Depending on the instantiated type of root derivation, the function chooses a different entry algorithm. Figure 4.12 demonstrates the lifecycle of such an example. Synchronous derivation *sd0* will be marked as `Requested` when the algorithm traverses, which serves as a signal when the algorithm determines the timeliness of downstream asynchronous derivation *ad2*. The agent monitoring *ad2* will request an update for *sd0* and start its execution after it confirmed that all its immediate upstream asynchronous derivations are up-to-date.

So far we have introduced three parts for DIF that work together in a cohesive fashion: the derivation ontology, the derivation agent, and the derivation client. The fundamental objective of the framework is to ensure that all provenance information is explicitly documented within the dKG, capturing details on what calculations were performed, their origins, and the functions (agents) employed. This allows for the temporal coherence, *i.e.*, the ordering of the timestamps of events. As such its correctness can be verified by the provided function `validateDerivations`, which ensures there are no circular dependencies, and each instance (pure inputs and derivations) has a valid timestamp. Additionally, we have included test cases to cover the key features and functionalities of the framework. There also exist minimal working examples in both Java and Python as tutorials for newcomers. For these resources, please refer to `TheWorldAvatar` repository open-souced on GitHub[3] and PyPI[4]. We believe this transparency facilitates verification of the accuracy of the developed framework.

[3]https://github.com/cambridge-cares/TheWorldAvatar

[4]https://pypi.org/project/pyderivationagent/

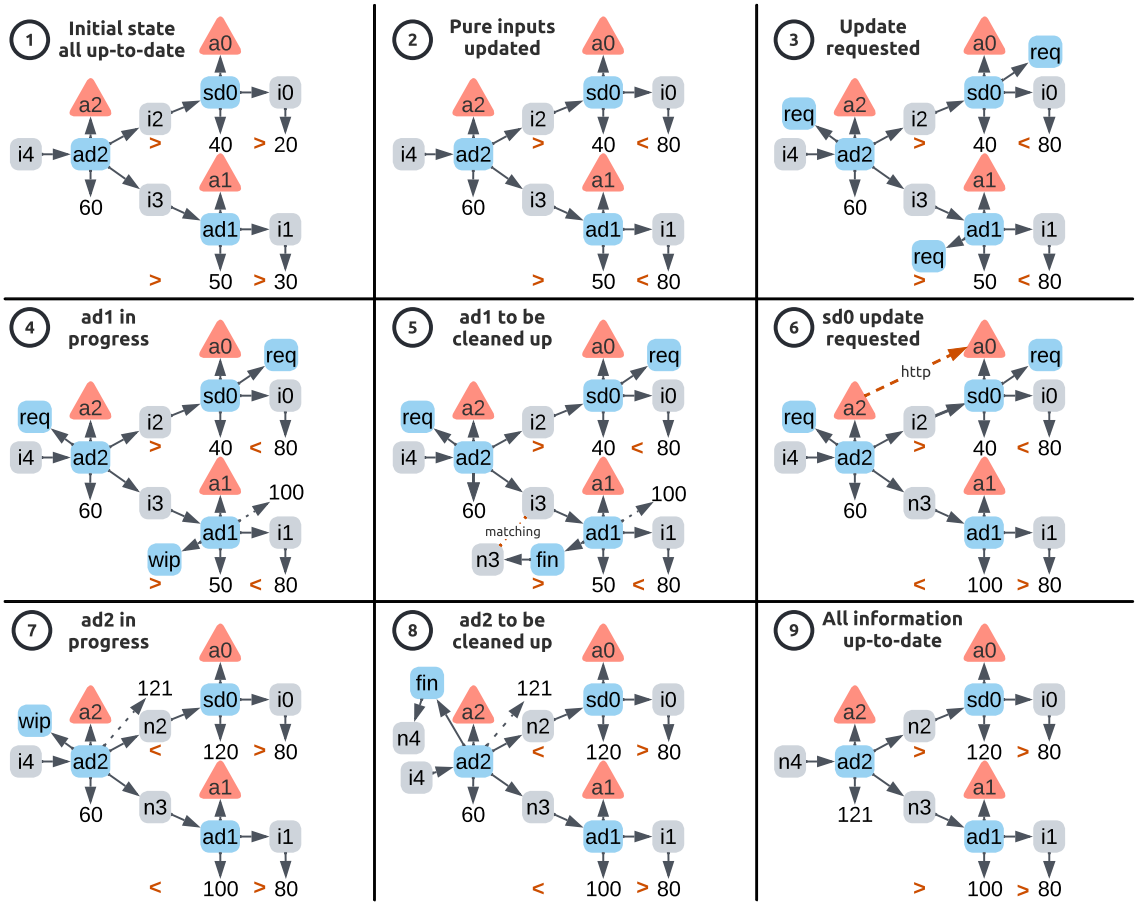


Fig. 4.12 The process of updating a derivation DAG consisting of asynchronous derivations that are dependent on synchronous derivations.

4.3 Results and discussions

Flood impacts can be classified as direct versus indirect and tangible versus intangible effects [144]. This work serves as a proof-of-concept for the automatic assessment of flood impacts focusing on direct tangible impacts, referring to assets at risk due to direct physical exposure to floodwater. The intangible and indirect impacts of a potential flooding event are not considered in this work.

The impact estimation considers the number of buildings and the total property value that are potentially at risk by a potential flooding event. This involves information and events that occur at varying rates. However, existing approaches primarily focus on static evaluations and scenario planning [144], lacking the capability to offer a real-time view of the value at risk. This limitation highlights the necessity for cross-domain interoperability to adequately address the dynamic nature of such events. As demonstrated in the previous section, DIF

semantically annotates the dependencies between different pieces of information and uses computational agents to keep them current as a living snapshot of real-world events, making it an ideal candidate to address this problem.

The details of the domain ontologies and agent logic can be found in [160]. Here, we focus on how DIF is used for this UK-based use case with a simplified example, including the coverage of the derivation subgraph and the processes that occur as information is cascaded over time. The results for the actual data are presented using the Digital Twin Visualisation Framework (DTVF) that is part of the World Avatar.

4.3.1 Automated population and update of the derivation subgraph

The flood impact assessment uses data from various sources, including APIs such as the Environment Agency Real Time Flood-Monitoring API [80], Energy Performance Certificates (EPC) [81], and HM Land Registry Open Data [158]. These data are instantiated and updated in the World Avatar dKG regularly by input agents. Using the source information, the impact assessment involves various derivation agents that work together to populate a derivation subgraph. This process encompasses two types of actions on the derivation subgraph: creating newly derived information, such as the impact of a newly issued flood warning, and updating existing information, such as the updated impact of an existing flood warning when some of the source information is updated.

One of these derivation agents, the Flood Assessment Agent, calculates the flood impact by identifying the buildings located within the affected area and determining the total market value of the properties at risk. The property value of each building is estimated by either scaling its most recent transaction record based on the local property price index or by multiplying its floor area by the average square metre price for its postal code. This representative average price can be computed by the Average Square Metre Price Agent considering all of the most recent transaction records in the area.

Figure 4.13 illustrates progresses in the derivation subgraph when evaluating the potential impact of an issued flood warning. As denoted by the red dashed arrow, the flood warning instance is instantiated by the Flood Instantiation Agent. It specifies information about the expected severity of a flood event and the specific geospatial extent that is at risk. In this simplified example, the geospatial polygon is assumed to cover a postal code that includes two buildings, each with data about its floor area and its most recent transaction record. Additionally, the UK property price index which captures changes in the value of residential properties is instantiated as a time series in the dKG and is updated on a monthly basis.

Once the derivation agents are deployed, the population of derived information starts with marking up the average square metre price derivation for all postcodes within the region

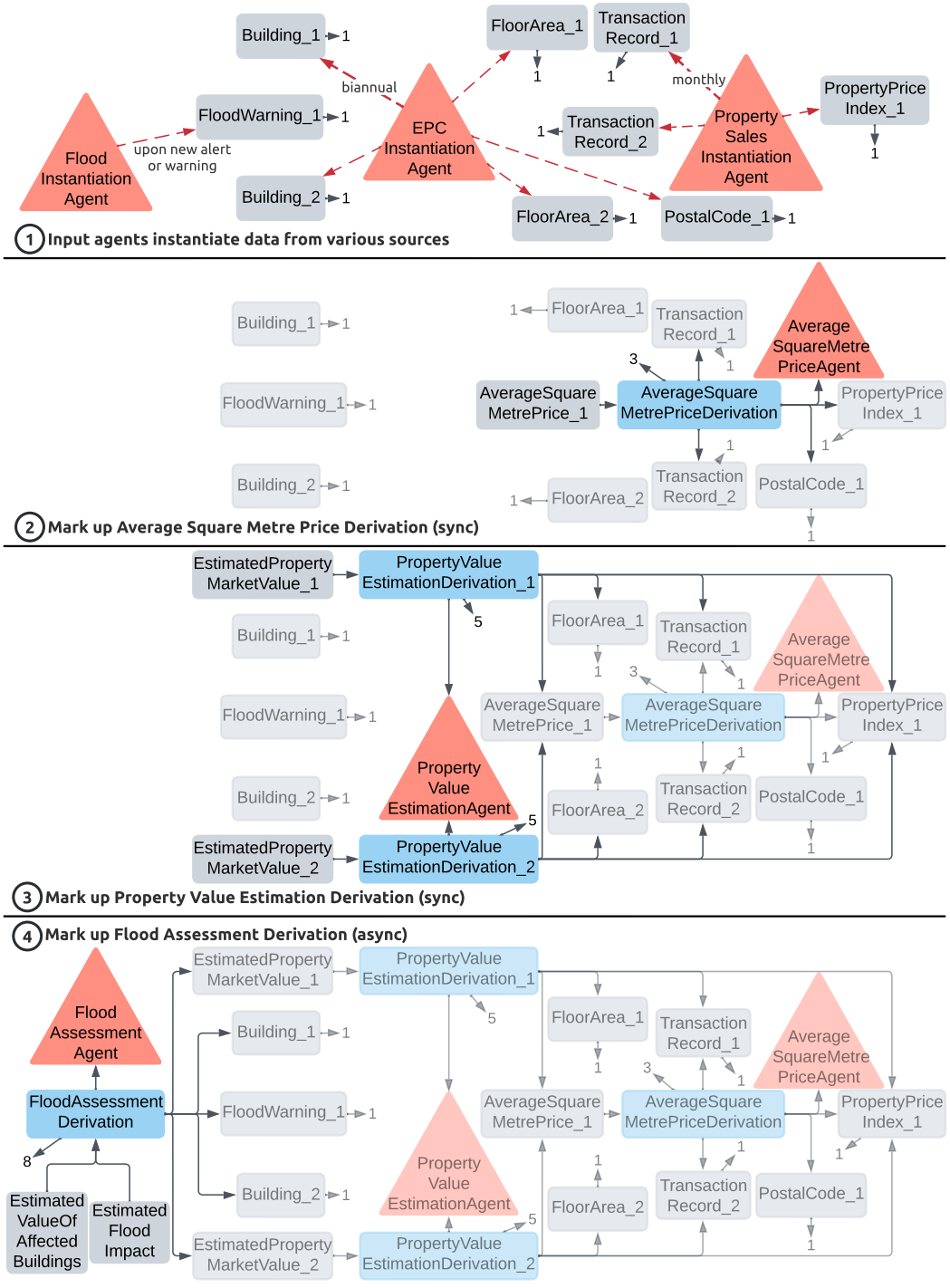


Fig. 4.13 The process of populating the derivation subgraph for flood impact assessment. Entities with low opacity represent existing entities in the dKG, *i.e.*, added in the previous steps of agents' operation before the agent adds newly derived information. The integers attached to the entities denote exemplary timestamps at which this information has been added to the dKG.

of interest. Next, the property value estimation derivation is marked up for all buildings. Since these computations are relatively fast, they are marked up as synchronous derivations to obtain instantaneous responses. The flood assessment derivation is marked up upon the instantiation of the flood warning instance. In practice, a flood warning can cover more than a dozen postal codes with hundreds of buildings. Its computation can take some time and is thus marked up as an asynchronous derivation. Consequently, the absence of a status associated with the flood impact assessment derivation indicates that the derived information is up-to-date, signifying the completion of the assessment process.

As each input agent operates at different frequencies, the impact of a flood can alter when the source information is updated while the warning is still active. For example, the property price indices for all administrative districts in the UK are updated monthly, and the geospatial extent of an active flood warning can also change, both of which can result in outdated flood impact assessments. Like the update of any other derived information, the impact assessment is designed to be on request to save computational power. Upon an update request, DIF traverses the derivation subgraph to check if the average square metre price is up-to-date and if not, it backtracks to mark the derivations as outdated which triggers the flood assessment as "requested". Subsequently, Flood Assessment Agent triggers an update by calling the Average Square Metre Price Agent directly, *i.e.*, synchronously. The price is updated in the dKG which is further utilised by the Property Value Estimation Agent to update the estimated property market value. Only when all the input information is updated, the Flood Assessment Agent starts the impact estimate. This guarantees the information used in the assessment faithfully reflects the real world. The use case demonstrates both communication modes of DIF by utilising both synchronous and asynchronous derivations.

4.3.2 Visualisation of potential flood impact

Following the simplified example, we present the visualisation of the impact assessment using real data. The derivation markup was created for a flood warning covering 34 postal codes and 289 buildings altogether. It is important to highlight that although there is no specific metric employed to directly assess whether all buildings within the flood area are accounted for by DIF, the coverage of inputs can be evaluated by comparing the buildings identified as at risk with the polygon of the potential flooding area via SPARQL queries. To ensure the accuracy of results, all agents developed in this work are tested on synthetic data. For the real-world building information used in the study, data were obtained from EPC and HM Land Registry Open Data. However, the extent of completeness in their respective database lies beyond the scope of this work.

Figure 4.14 visualises the estimated impact of a potential flood event. The representation separates different data into distinct layers, including buildings and the geospatial area affected by the active flood warning. By overlaying the flood boundary on the map, it is possible to identify which properties are at risk. The buildings within the flooded region are colour-coded with their estimated property values, while the other buildings in the administrative district are included only with their location information.

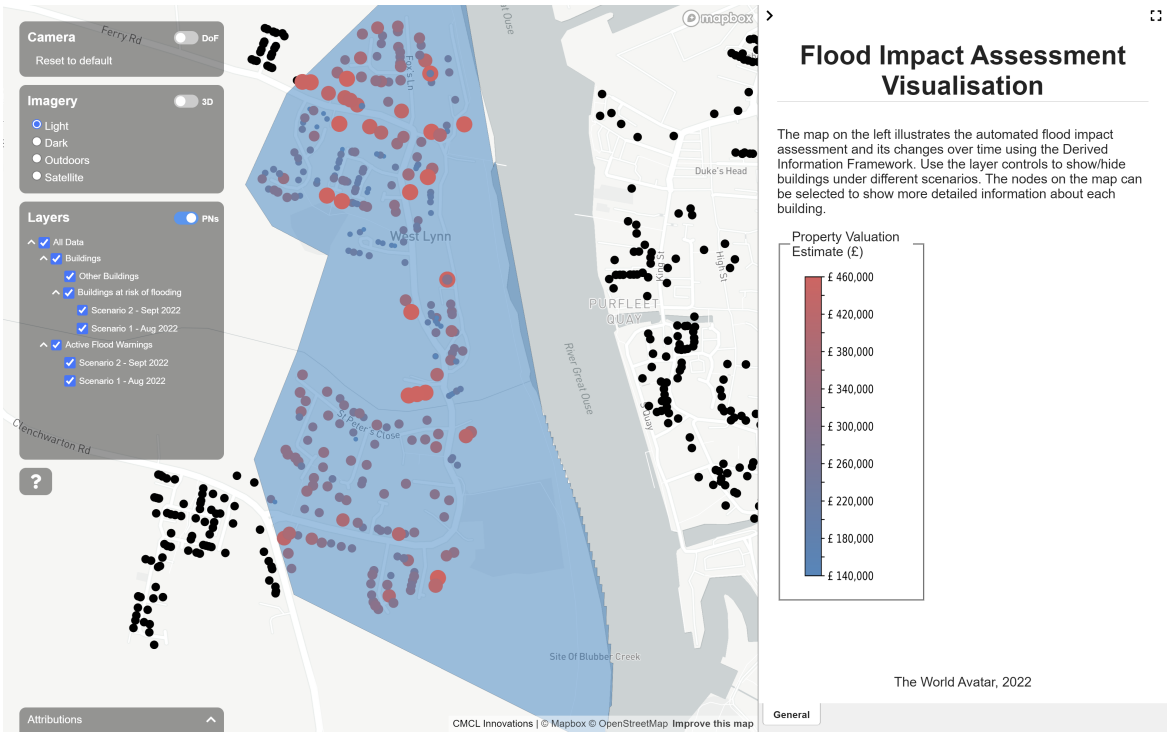
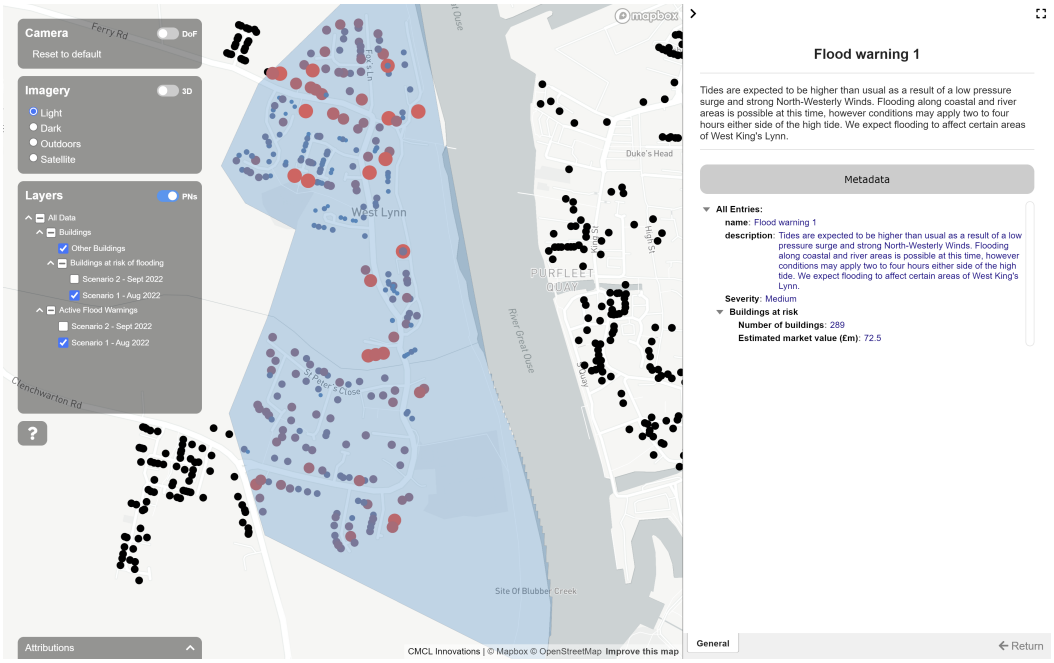
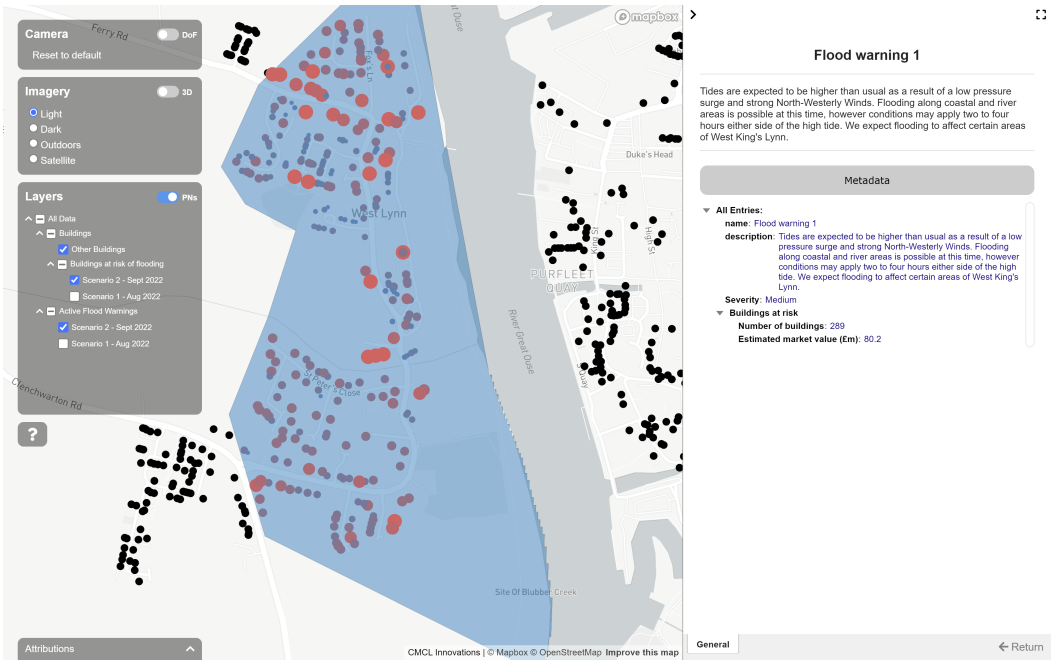


Fig. 4.14 Web page visualising a flood impact assessment with layers for buildings and flood warnings. The blue region denotes a potential flooding area. The colour and size of the dots in this region reflect estimated property values. Buildings outside of the region are considered unaffected and are thus marked with black dots.

Figure 4.15 presents two scenarios that are available to be selected on the plot: the estimated flood impact in August and September 2022. Between these two scenarios, an update of the district's property price index has been factored into calculating the total property value at risk. When clicking on the flooding area in different scenarios, the estimated impact and the detailed description of the flooding event are queried by the DTVF on-the-fly and displayed on the side panel. As the derivation agents manage the dKG, the visualisation will be automatically updated.



(a) Scenario 1: Impact assessment of a newly issued flood warning.



(b) Scenario 2: Impact update of an existing flood warning after property price index has increased.

Fig. 4.15 Automated flood impact assessment and update using DIF. The side panel displays information about the clicked feature that has been dynamically retrieved from the dKG.

4.3.3 Scalability in various scales of flooding events

To examine the performance and scalability of DIF in handling various scales of flooding events, tests were carried out on a virtual machine hosted on DigitalOcean. The virtual machine features 4 Intel Xeon Gold 6248 CPUs operating at 2.50 Ghz and 32 GB DIMM RAM. The triplestore and agents are deployed as individual docker containers. The analysis consists of two cases: (1) the instantiation of new flood warnings only, and (2) the update of the property price index, followed by the instantiation of new flood warnings. The second case triggers updates for the average square metre price and estimated property market value of all buildings potentially at risk. Tests for both cases were performed for three flood warnings each covering a different number of buildings at risk: 398, 920 and 5147. We measure from the time the framework picks up the derivation for flood impact (re-)assessment to the timestamp when all information is up-to-date. Each test was repeated at least three times to obtain the mean and standard deviations in processing time.

Figure 4.16 illustrates the linear scalability of DIF across the varying number of buildings examined. It should be noted, however, that the actual amount of processing time is use-case-specific. By deducing the processing time of case 1 from case 2, it is observed that the agents process approximately four buildings' derivations per second. At present, only one instance for each agent is deployed. In order to evaluate the performance gains achievable through the deployment of multiple instances of a given agent, the introduction of a load balancer becomes necessary which is however beyond the scope of this work.

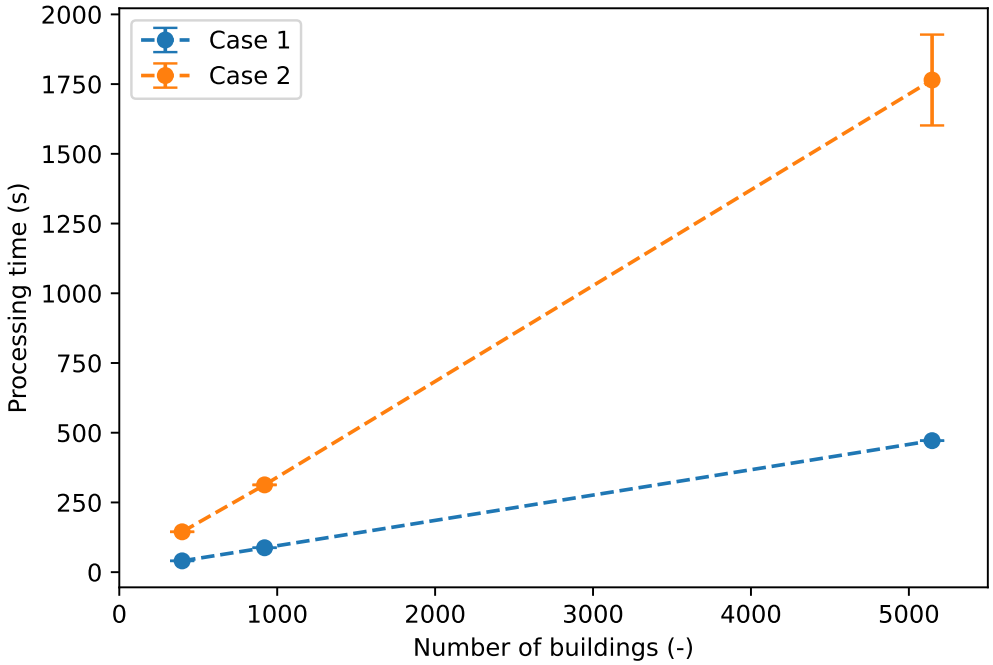


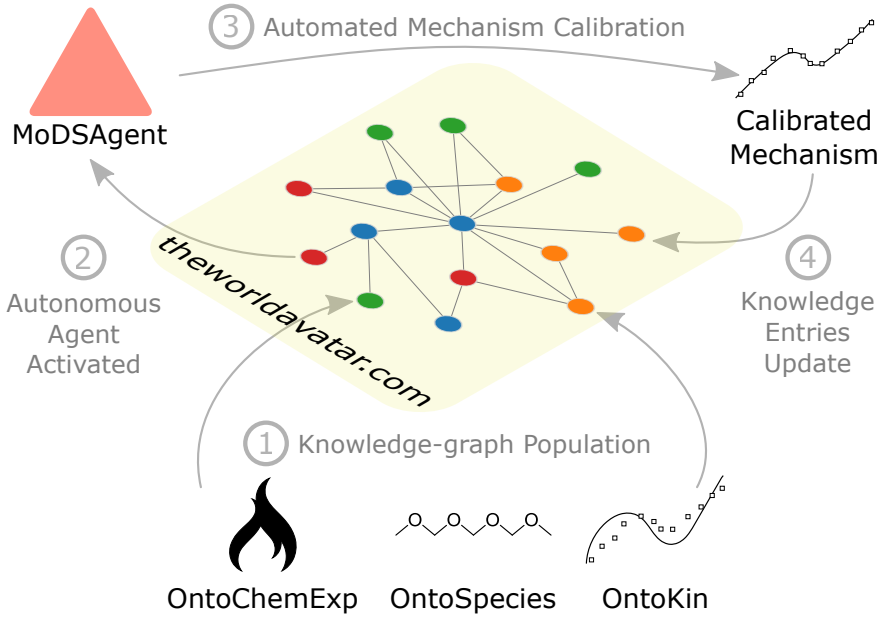
Fig. 4.16 Processing time when DIF updates flood impact assessment for potential flooding events with different amounts of buildings at risk. Case 1: the instantiation of new flood warnings only. Case 2: the update of the property price index, followed by the instantiation of new flood warnings.

4.4 Chapter summary

In this chapter, we developed DIF as a KG-native solution for tracking data provenance and managing information flow within a dKG. It expands previous capabilities and further abstracts complexity away from the developers of individual agents. The architecture includes a lightweight ontology that marks up agent communication as provenance records in the KG, an agent template that standardises the operation of agents in both synchronous and asynchronous communication modes, as well as a client library that offers functions for managing the derivation subgraph. The framework demonstrated its capability via a flood impact assessment use case that automatically populates the derivation subgraph and refreshes the derived information upon request when new information arrives. The knowledge dynamics were enabled by means of agents' actions while preserving the provenance information. This chapter laid the groundwork for automated workflow management which is an essential component for developing connected SDLs.

Chapter 5

Automated mechanism evaluation and calibration



"Any agreement between model and experiment is sheer luck."

– David Golden, as quoted by Michael Frenklach in Transforming Data into Knowledge–Process Informatics for Combustion Chemistry (2007), Proceedings of the Combustion Institute, 125–140

The chapter draws from a paper published in *Journal of Chemical Information and Modeling* in collaboration with the Computational Modelling Group at the University of Cambridge. Mr Geeson and Dr Bringley assisted in the preparation of the simulation script for *k*inetics and Cantera, respectively. Dr Farazi advised the creation of ontology and agent. All authors provided feedback. The ontology/agent development, simulation, and writing were performed by the author.

In this chapter, the dKG approach is applied to automate computational model development performed on an HPC facility. Specifically, the study focuses on the automated calibration of combustion reaction mechanisms and demonstrates its effectiveness in a case study of poly(oxymethylene) dimethyl ether ($PODE_n$, where $n = 3$) oxidation. OntoChemExp is developed as an ontological representation for combustion experiments that allows for the semantic enrichment of experiments. Following this, a set of software agents are developed to perform experimental results retrieval, sensitivity analysis, and calibration tasks. The sensitivity analysis agent is used for both generic sensitivity analyses and reaction selection for subsequent calibration. The calibration process is performed as a sampling task followed by an optimisation task. The agents are designed for use with generic models but are demonstrated with ignition delay time and laminar flame speed simulations. Compared to manual calibration, this work has reduced the calibration time from months to days with a 79% decrease in objective function value as defined in this chapter.

5.1 Introduction

The contribution of human activity to climate change and the potential for ecological devastation this presents has become a widely accepted fact within the scientific community [193]. One of the key contributors to this effect is the release of greenhouse gases from combustion processes. Improvements in the design of energy conversion systems have already resulted in significant reductions in their contribution to greenhouse gas emissions and present one of the potential paths towards even lower emissions in the future. Another approach involves the use of alternative fuels, particularly synthetic fuels, offering the potential for reductions in pollution and greenhouse gas emissions.

Modern workflows for the design and optimisation of combustion equipment and devices now routinely employ computational modelling techniques. These are most often used to screen designs and offer invaluable insights into the chemical processes [122]. Thus, to achieve the desired reduction in climate change potential from combustion equipment, the provision of accurate combustion chemistry mechanisms is becoming essential.

In practice, the development of these combustion chemical mechanisms consists of two parts: mechanism generation and mechanism calibration. The first step constructs a tentative mechanism that maps the reaction pathways via elementary reaction generation and selection, and the second step adjusts the rate parameters, attempting to faithfully reproduce experimental observations.

Much of the construction of combustion mechanisms involves the selection and combination of reaction families, elementary reactions and sub-mechanisms from various existing mechanisms. This is facilitated by the CHEMKIN [185] mechanism format, acting as a *de facto* standard for mechanism sharing within the combustion community. Aspects not formally captured by this format include semantics and provenance. This allows errors to propagate through combustion models due to the inability to ensure the quality of individual reactions and the difficulty of tracking their origin.

Moreover, the accuracy and consistency of combustion mechanisms are not guaranteed across applications, even with well-calibrated sub-mechanisms [111]. These problems are further exacerbated when the scale of the mechanisms is considered; potentially hundreds of species and thousands of reactions may be involved. This leads to the manual curation and provenance determination of all the components of these mechanisms being a near-impossible task for researchers. Even if attempts were to be made, these are likely to fall foul to human error and so a very real need for an automated approach to this mechanism development task is present.

Reaction Mechanism Generator (RMG) [115] is one of the available tools for the automation of the first stage of mechanism development. The technique is based on the use of a set of chemical rules to predict chemical pathways along with a database of chemical properties. Values of unknown chemical properties are estimated on-the-fly. One of the methods used for this purpose is that of Li et al. [211], employing a Graph Neural Network (GNN) on molecular graphs to make formation enthalpy predictions. The GNN was trained on a dataset generated at B3LYP/6-31G(2df,p) level of DFT. The framework further incorporated quantum chemistry calculations for additional model training in case of uncertain predictions, improving accuracy and generalisability.

Progress has also been made in automating transition state theory calculations [22], an important step towards generating accurate reaction rates. However, generating a chemical mechanism with purely quantum-calculated rate parameters remains infeasible, given that even the most detailed model would not include all possible pathways [193]. This necessitates the use of automated calibration processes for these coefficients to reproduce experimental results.

The mechanism development and curation process may be improved in two ways: (1) semantically focused and machine-interpretable formats for mechanism representation with clear provenance should be used and (2) automated updating and verification of existing mechanisms throughout their lifetimes should be implemented. Task 1 has received attention within the community, with various efforts to create standardised databases of combustion data with unique identifiers and easily processable formatting. One of the key efforts in this direction is that of the Process Informatics Model (PrIMe) [110] database, containing combustion data in a standardised XML format. Varga et al. [366] further developed the ReSpecTh Kinetics Data (RKD) format which is an extended version of the PrIMe XML format with new elements for unique identification of experimental setup and data. Computational packages, for example, Optima++ [367], are also provided alongside RKD for carrying out simulations and interpreting experimental results. ChemKED [378] and its related Python-based package is another effort to provide a standard format for experimental data in the combustion community.

Although projects such as PrIMe have started the process, further strides towards a fully provenanced and machine-interpretable standard for the combustion community must continue. The relatively granular structure of databases and the lack of semantics prevent them from reaching the true potential of modern technologies within AI and knowledge discovery. One of the potential technologies to aid with these processes is dKG, a dynamic knowledge ecosystem that interconnects information and software agents. The implementation employs ontologies to define concepts and relations that are shared within the community. Such a design offers clear semantics to its entries with a highly linked structure, enabling item locating, provenance determination, and reasoning over entries with software agents.

The purpose of this chapter is to demonstrate the dKG approach for computational model development, specifically the automated combustion mechanism calibration. This forms a clear path for achieving the second of the highlighted tasks. We aim to achieve this by constructing an ontology to link combustion experiment measurements to chemical reaction mechanisms and developing a set of software agents that automatically perform mechanism calibration against ignition delay time and laminar flame speed experimental data. A demonstration of this is performed on a reduced Poly(Oxymethylene) Dimethyl Ether 3 ($PODE_n$, where n=3) mechanism [213]. This alternative fuel, with a molecular formula of $CH_3O(CH_2O)_3CH_3$, is deliberately chosen for its current interest as a fuel additive to help with the push for cleaner and more efficient vehicles. The additive has been demonstrated to lower soot emissions and improve combustion efficiency in engines [215]. To match the current interest and to further $PODE_3$'s commercialisation, a reduced yet robust mechanism is required, making it an ideal candidate for a demonstration of our framework.

5.2 Methodology

5.2.1 Mechanism calibration

PODE demonstration case and simulation procedure

This case serves as a demonstration of the developed framework. The particular case of $PODE_3$ has received recent interest and a range of previous efforts for modelling its combustion processes accurately when used as part of a fuel blend. Some of the prior works in this area are listed in Table 5.1, with many of the mechanisms intended for Primary Reference Fuel (PRF) blends.

Table 5.1 Summary of existing $PODE_n$ ($n = 2, 3, 4$) combustion mechanisms with their statistics counted in the OntoKin format. The reduced mechanism developed in Lin et al. [213], before optimisation, is chosen as the starting mechanism for the demonstration case of the dKG-based automated mechanism calibration approach proposed in this chapter. It should be noted that the number of reactions of the OntoKin representation is different to that of CHEMKIN format.

Mechanism	Type	No. Species	No. Reactions	Fuel	Fuel Carrier
Sun et al. [348]	Detailed	274	1674	$PODE_3$	No
He et al. [151]	Detailed	225	1178	$PODE_3$	No
He et al. [150]	Detailed	354	1392	$PODE_3$	Yes, PRF
Ren et al. [294]	Reduced	145	668	$PODE_3$	Yes, PRF
Lin et al. [213]	Reduced	61	215	$PODE_3$	Yes, PRF
Cai et al. [38]	Detailed	322	1611	$PODE_{2–4}$	No

The first mechanism for pure $PODE_3$ combustion under high-temperature conditions was developed by Sun et al. [348]. This was an example of a detailed combustion mechanism, whereby an attempt is made to model all elementary reactions believed to be present. In contrast, reduced combustion mechanisms are constructed to replicate results of key combustion metrics with a reduced set of reactions.

Following from the Sun et al. [348] mechanism, low- and intermediate-temperature conditions were covered by He et al. [151]. This model was further expanded by He et al. [150], which is the first-ever mechanism to describe the combustion of a $PODE_3$/PRF blend. Given the complexity of engine simulations, the size of these mechanisms makes simulation largely intractable, requiring the development of a reduced mechanism. Two simplified mechanisms were proposed by Ren et al. [294] and Lin et al. [213]. Both employed the model of He et al. [151] as a basis, using different methodologies for selecting key species and reactions of $PODE_3$. Additional reactions were added by both for modelling the combustion of a PRF carrier fuel.

A notable alternative attempt was made in the work of Cai et al. [38]. In this, an automated mechanism development process is used to select the reactions for the detailed combustion mechanism of $PODE_n$ ($n = 2, 3, 4$). The work employed the CLass-based Automatic Reaction Alternator (CLARA) and calibrated the selected reactions against experimental data for the ignition delay of $PODE_n$/air mixtures.

As a demonstration of the dKG approach, the starting mechanism of Lin et al. [213] is selected due to its relatively small size. This is the mechanism generated by selecting reactions from the wider He et al. [151] mechanism prior to any further calibration of experimental results. The focus of this work remains the calibration of the $PODE_3$ combustion mechanism and so only $PODE_3$ combustion experiments are chosen for calibration.

The calibration was carried out against Rapid Compression Machine (RCM) ignition delay time [151] and laminar flame speed [348] experiments. The ignition delay times of $PODE_3/O_2/N_2$ mixtures were measured at pressures of 10 bar and 15 bar, over a temperature range of 641 K to 865 K, with equivalence ratios of 0.5 (O_2:$N_2 = 1:8$), 1.0 (O_2:$N_2 = 1:15$), and 1.5 (O_2:$N_2 = 1:20$). The laminar flame speeds of $PODE_3$/air mixtures were measured at atmospheric pressure and an initial temperature of 408 K, with equivalence ratios ranging from 0.7 to 1.6. For the simulation stage, the ignition delay time is defined as the time interval between the starting point and the maximum rate of pressure rise due to the ignition. The laminar flame speeds were calculated with a mixture-averaged transport model.

A core strength of dKG is its ability to combine data and software from different sources in a standardised way, achieving interoperability and extensibility. In the present application, this means the ability of a generic tool for calibration of any grey- or black-box model to deal with a variety of computational software. As a first step towards this goal, different modelling software for ignition delay times and laminar flame speeds are employed to demonstrate the competence of the framework at handling models in a generic manner. Specifically, the simulations were performed using *k*inetics (version 2020.1.1) [56] for ignition delay times and Cantera (version 2.4.0) [131] for laminar flame speeds. For the laminar flame speed simulations, Soret effects were not considered and the solution gradient and curvature were both fixed at 0.05. The grid was set to be refined with a pruning coefficient of 0.01. The simulation scripts were prepared by the author with help from Mr Geeson for *k*inetics and Dr Bringley for Cantera.

Sensitivity analysis

Sensitivity analysis acts as a screening process to identify reactions that have measurable effects on the model responses [109]. This is conducted by computing the normalised

sensitivity coefficient of the chosen response with respect to the Arrhenius pre-exponential factors of the starting mechanism.

At the n^{th} point in the process condition space $\xi^{(n)}$, the normalised sensitivity coefficient of the i^{th} response $\eta_i(\xi^{(n)},\theta)$ with respect to the j^{th} model parameter θ_j is defined as [364]:

$$\frac{\theta_j}{\eta_i(\xi^{(n)},\theta)}\frac{\partial\eta_i(\xi^{(n)},\theta)}{\partial\theta_j}. \tag{5.1}$$

Due to the complexity and stiffness of a typical combustion mechanism, a numerical solution is normally adopted [364]. The vector showing a small relative perturbation r of model parameters in the j^{th} positive direction can be denoted as:

$$\tilde{\theta}^j := (\theta_1,\ldots,\theta_{j-1},(1+r)\times\theta_j,\theta_{j+1},\ldots,\theta_n), \tag{5.2}$$

yielding a finite difference approximation of the normalised sensitivity coefficient as:

$$\frac{\theta_j}{\eta_i(\xi^{(n)},\theta)}\frac{\eta_i(\xi^{(n)},\tilde{\theta}^j)-\eta_i(\xi^{(n)},\theta)}{(\tilde{\theta}^j-\theta)_j}=\frac{\eta_i(\xi^{(n)},\tilde{\theta}^j)-\eta_i(\xi^{(n)},\theta)}{r\eta_i(\xi^{(n)},\theta)}. \tag{5.3}$$

Considering the sensitivities across the entire range of process condition space N, the overall sensitivity S_{ij} of the i^{th} response with respect to the j^{th} model parameter can be determined either by a maximum absolute value:

$$S_{ij}=\max_n\left\{\left|\frac{\eta_i(\xi^{(n)},\tilde{\theta}^j)-\eta_i(\xi^{(n)},\theta)}{r\eta_i(\xi^{(n)},\theta)}\right|\right\}, \tag{5.4}$$

or an averaged absolute value:

$$S_{ij}=\frac{1}{N}\sum_N\left[\left|\frac{\eta_i(\xi^{(n)},\tilde{\theta}^j)-\eta_i(\xi^{(n)},\theta)}{r\eta_i(\xi^{(n)},\theta)}\right|\right]. \tag{5.5}$$

It should be noted that this analysis is *local* in the sense of model parameters while *global* in the process condition space, such that sensitivities at every collected point in the experiment are taken into account.

As chemical mechanisms can either be assembled from reaction classes or individual elementary reactions, it is natural to either optimise reactions on a class basis or just based on individual reaction's contributions. Cai and Pitsch [37] demonstrated a comparable performance between both bases, claiming that a significant distinction would only appear

when reactions in the same class show low sensitivity individually but high sensitivity collectively.

In the case of a combustion mechanism constructed for a group of similar species, optimisation based on reaction rules often provides a consistent improvement of model performance. This was found to be the case with the mechanism developed in Cai et al. [38], describing $\text{PODE}_{2\text{–}4}$ combustion. The comparable performance is seen as justification for implementing only one of the bases at present. The selected basis is that of individual reaction contributions, chosen because many of the envisaged use cases will involve only one or few key staring species. Further option for calibration on a class basis will be implemented in future work.

Global search and local optimisation

In order to find an optimal balance between the two considered responses, a weighted least-squares objective function was implemented for the experiment responses:

$$\begin{aligned}\Phi(\theta) = &\sum_{n=1}^{N}\left[[\eta_{\text{ign}}(\xi^{(n)},\theta)]' - [\eta_{\text{ign}}^{\text{exp}}(\xi^{(n)})]'\right]^2 \\ &+ \alpha \times \sum_{m=1}^{M}\left[[\eta_{\text{fls}}(\xi^{(m)},\theta)]' - [\eta_{\text{fls}}^{\text{exp}}(\xi^{(m)})]'\right]^2,\end{aligned} \tag{5.6}$$

where α refers to the weighting of laminar flame speed in the calibration process.

Following selection of the target reactions through sensitivity analysis, an optimisation routine is followed to calibrate the mechanism with the objective function defined above. The process initially employs low-discrepancy quasi-random global sampling through a Sobol sequence generator [337]. This provides initial points for a Hooke-Jeeves optimisation algorithm [164], selected for its gradient free nature to better handle the stiff system.

In each evaluation, experiment and model responses are scaled with respect to the upper η_{ub} and lower η_{lb} bounds of the experimental observations, as defined by the experimental unceratinty. For ignition delay times, a $\pm 20\%$ uncertainty was assigned to the measurements. This was selected to align with common practices within the community [36, 175, 38].

As ignition delay times can vary by orders of magnitude, a logarithmic scaling was applied to balance the contribution of each data point towards to the objective function:

$$\eta' = \frac{\log(\frac{\eta^2}{\eta_{\text{ub}}\eta_{\text{lb}}})}{\log(\frac{\eta_{\text{ub}}}{\eta_{\text{lb}}})}. \tag{5.7}$$

For laminar flame speed data, the error used was that reported by the source [348]. A linear scaling was applied:

$$\eta' = \frac{2\eta - (\eta_{ub} + \eta_{lb})}{\eta_{ub} - \eta_{lb}}. \tag{5.8}$$

Uncertainty bounds may be obtained from Uncertainty Factors (UF) for Arrhenius rate equation parameters [328], derived from the uncertainties in quantum chemistry calculations. There are also alternative optimisation principles for the reactions involved in combustion chemistry that optimise both activation energies and pre-exponential rate parameters in a coupled manner [10] as well as techniques that include the temperature dependence exponent.

At present, the optimisation of just pre-exponential factors has been performed. This was chosen to simplify the process for a proof-of-concept as the main focus of this chapter is to demonstrate the utility of a dKG in computational model development. There is an existing precedent for works adjusting the pre-exponential factors alone [213], with large hypercubes of potential values. This approach is further justified when reduced mechanisms are optimised as some reaction pathways and species are already not present within the mechanism. It should be noted that other optimisation techniques can be easily made available in future work.

5.2.2 Ontological representation

Dr Farazi assisted the author in the creation of the OntoChemExp ontology. It is developed to describe combustion *experiments*, detailing both the overall experimental setup and the individual, independent variable values for each data point. The overall experimental description of the ontology incorporates the *apparatus* used and the various *common properties* shared amongst data points. Independents are used to form *data groups* that share the same set of independent variables, with individual data points forming subsets of these data groups.

The current structure of OntoChemExp is developed following discussions with domain experts and takes inspiration from existing databases, including the experimental data stored in the PrIMe database. The complete ontology contains 36 concepts and 60 relations. OntoChemExp is available on GitHub[1].

Figure 5.1 illustrates the core concepts and relations defined in OntoChemExp. The conceptual structure is divided into four modules following a heuristic approach:

- **Experiment**: An *experiment* refers to the process of observing and measuring characteristics of interest from an energy-release chemical process of a mixture of fuel and

[1]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontochemexp/OntoChemExp.owl

air. Dependent upon the original source, a set of metadata may be employed to provide more details and more precise identification of an experiment. This metadata includes *copyright*, *bibliography link* and *additional data item*.

- **Setup**: The setup outlines the global conditions of an experiment, including the *apparatus* in which the experiment was conducted, and the shared process conditions, forming *common properties*. The concepts defined in this section are normally left unchanged throughout an experiment.
- **Results**: Experimental *results* are grouped within *data groups* that share the same set of independent variables. Within the data groups, individual *data points* describe each experimental measurement, including independent and dependent variable values that are detailed within *X*.
- **Specification**: The specification is a shared data structure, supporting both the Setup and Results sections with an abstract concept *property*. Property is used to group a wide range of properties. The most straightforward usage is detailing the size of equipment with *value*. Property is also used to provide information about chemical *components* with the species described by a *species link* and an *amount*, *e.g.*, initial composition of fuel/air mixture. A further use of property is describing *derived properties*, that include *features* such as *indicators* and *observables*.

As an example, consider the laminar flame speed experiment conducted by Sun et al. [348]. The **Experiment** module contains the metadata related to this *experiment*, including a *bibliography link* that points to the publication and an *additional data item* specifying the description of the nature of the experiment. The **Setup** module documents the *apparatus* employed, *i.e.*, an electrically-heated constant-volume cylindrical combustion vessel, as well as the *common properties* that outline the boundary conditions used in the experiment for all data points, *i.e.*, a mixture of $PODE_3$/air at atmospheric pressure and an initial temperature of 408 K. This information was classified following the schema in Fig. 5.1 and detailed in the **Specification** module, *e.g.*, the mixture of $PODE_3$/air is represented by an initial composition *property* grouping *component* of individual chemical species with *species link* providing unique species identification and *amount* indicating concentrations. The **Results** module records the *data points* collected from the laminar flame speed measurements at equivalence ratios ϕ ranging from 0.7 to 1.6 in the format of *data groups*. This is also supported by the **Specification** module, such that both laminar flame speed measurement and its corresponding equivalence ratio were treated as individual *properties* that maps the human-readable notations (*e.g.*, commonly used mathematical symbol, units, description of this property, *etc.*) and the numerical values (*i.e.*, *X* as adopted in *data points*).

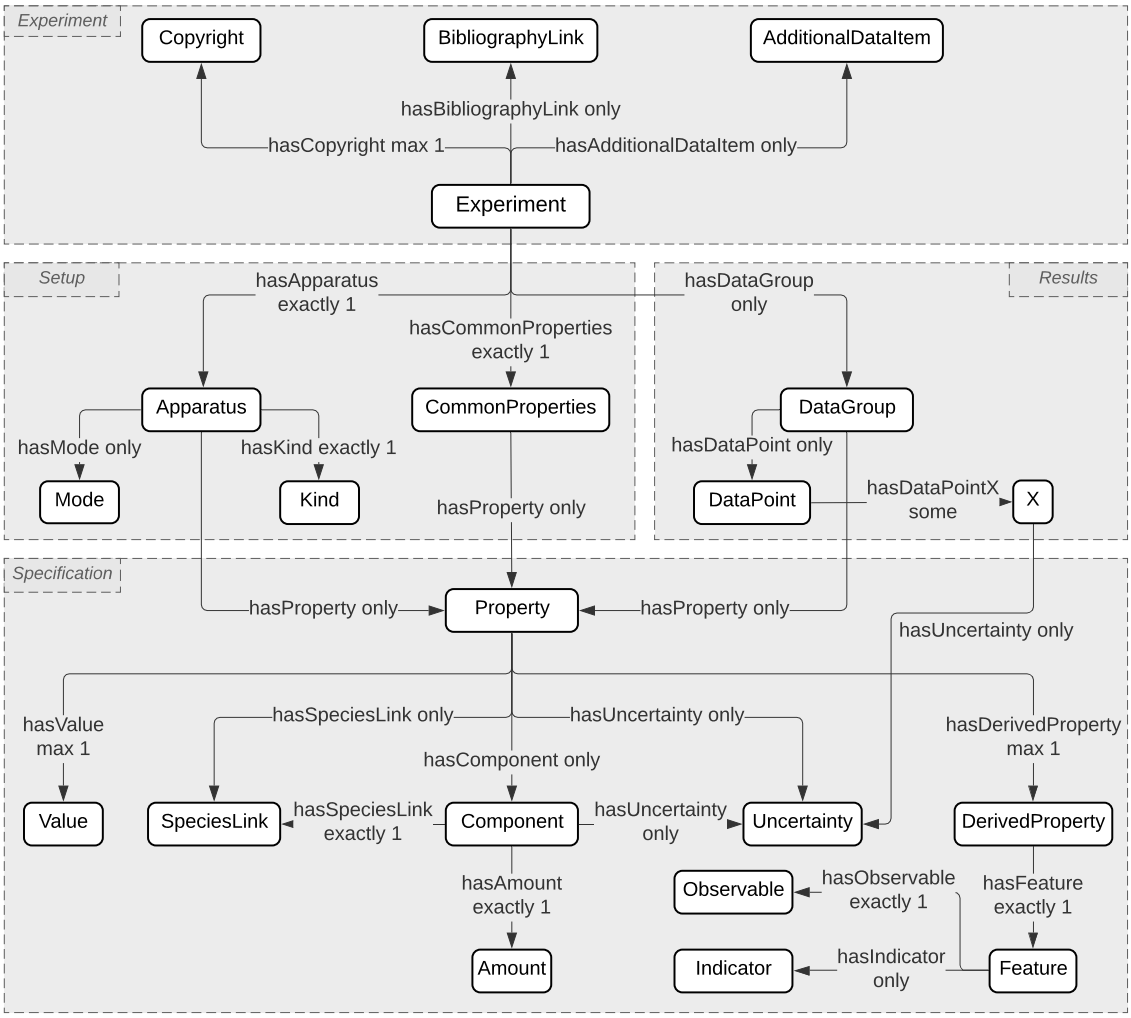


Fig. 5.1 Core concepts and relations of OntoChemExp ontology. This ontology is constructed to represent measurements from combustion experiments. The complete ontology consists of 36 concepts and 60 relations.

Figure 5.2 depicts how combustion experiment measurements and chemical mechanisms may be connected. The task of linking species with reactions has already been achieved in previous work [100], linking OntoKin with OntoSpecies. This allows the linking of OntoChemExp to both species and reactions via the provision of unique species identifiers within OntoSpecies.

Two connections are made between OntoChemExp and the prior ontologies: (1) data property `hasPreferredKey`, equivalent to `skos:altLabel`, that refers to the common name of a species within the community and (2) data property `hasUniqueSpeciesIRI` that directly links to the exact OntoSpecies instance. This design can help resolve inconsistencies between

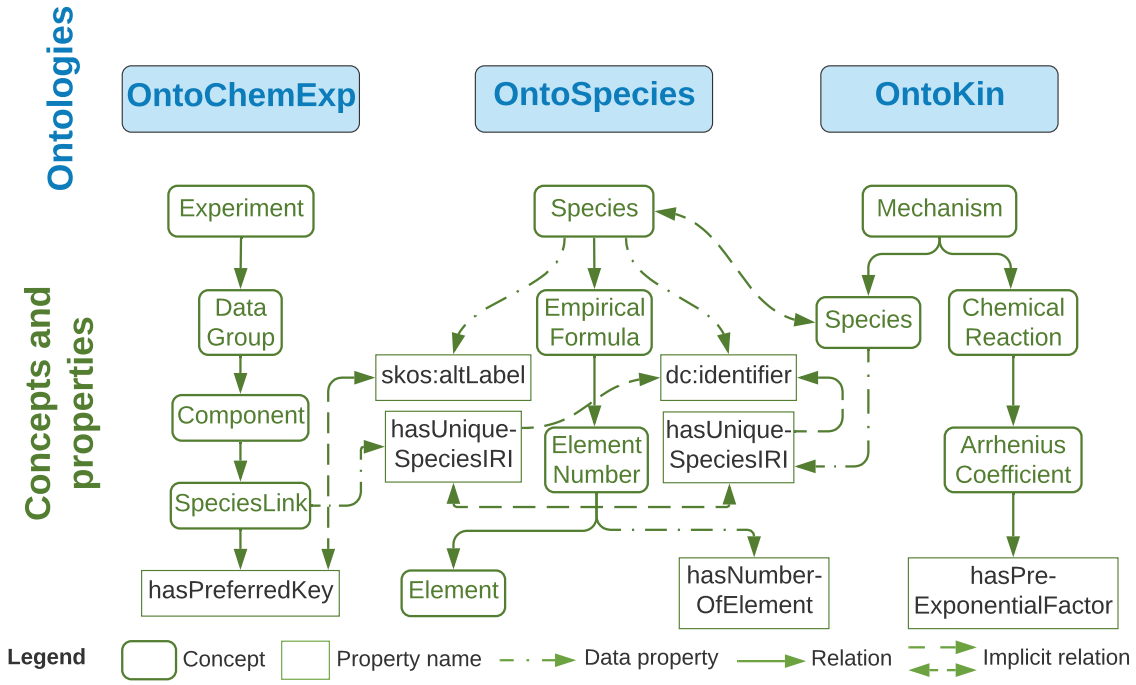


Fig. 5.2 Selected concepts, properties, and relations that demonstrate the links between OntoChemExp, OntoSpecies and OntoKin ontologies. The main purpose of these links is to enable unique identification of species.

data from different sources, through the unique identification of chemical species in OntoSpecies. The importance of the use of this approach is shown by the $PODE_3$ demonstration case whereby poly(oxymethylene) dimethyl ether 3 is denoted differently as $PODE_3$ by He et al. [151], $POMDME_3$ by Sun et al. [348], DMM_3 by Lin et al. [213], and OME_3 by Cai et al. [38]. These ambiguities may be handled by human operators but present a significant challenge to machine interpretability. This challenge is exemplified if $PODE_3$ is used as the notation for the initial concentration of ignition delay measurements and $POMDME_3$ for the laminar flame speeds experiment, whereas DMM_3 is used in the mechanism. Instead, the linkage is created between ontologies via unique species identification, such that one and the same species can be referenced throughout the various stages of calibration, irrespective of the different string labels that may be attached to it (which can be retrieved via SPARQL query) in different contexts. The ontological approach adopted thus facilitates dealing with naming ambiguities of chemical species, allowing for greater interoperability between agents, more comprehensive querying, and many opportunities for semantically driven tasks.

Populating the dKG is managed by a tailor-made toolset developed in this work for generating OntoChemExp-conformant OWL files. The toolset is based on that developed by

Farazi et al. [98] for converting chemical mechanisms to the format of OntoKin. Experimental data related to $PODE_3$ were manually constructed in the OWL format of OntoChemExp and then uploaded to the dKG. The dKG was subsequently deployed on an RDF4J[2] triple store, queriable by the SPARQL.

5.2.3 Agent integration

Dr Farazi advised the author in the development of the agents in this work. It is structured as an instantiation of the agent template proposed by Mosbach et al. [248]. A UML activity diagram of the agent is provided in Fig. 5.3, with the agent template surrounding the Model Development Suite (MoDS) [57] software package. MoDS is an integration of multiple tools developed for various generic model development tasks, such as parameter estimation [181], surrogate model creation [330, 394], and design of experiments [246].

Detailed documentation of the agent template is provided by Mosbach et al. [248] so only the changes from the template design will be detailed. One such change is the addition of a validation step for received job requests. This is added in order to improve the robustness of the agent. The validation step ensures that the job request to the sensitivity analysis agent contains the IRIs that point to the chemical model and the experiment data against which the sensitivity analysis is conducted. In addition, the job request to the mechanism calibration agent must contain the IRIs provided by either the user or the sensitivity analysis agent pointing to the active parameters to be optimised. The second change has been made to merge the process of querying executable-specific information from the dKG with the process of creating job files. This was implemented to accommodate different types of jobs being requested due to the integration of MoDS with multiple tools and its capability as a generic model development tool [246, 181, 247]. This results in an agent capable of automatically generating specific job files corresponding to supplied job requests. Once passed the request validation, the agent will query model parameters and experiment process conditions in the case of sensitivity analysis. By contrast, for mechanism calibration, additional queries are made for the list of active parameters and experimental responses.

The MoDS agent is designed to accept a target mechanism and experimental results at a range of process conditions and to perform parameter estimation for the target mechanism. To achieve this, the agent performs simulations with the experimental conditions and adjusts parameters within the target mechanism to replicate the experimental responses. The responses cover different simulation tasks which are performed in two different software packages, necessitating the generation of individual executable models. The software packages were

[2]https://rdf4j.org/

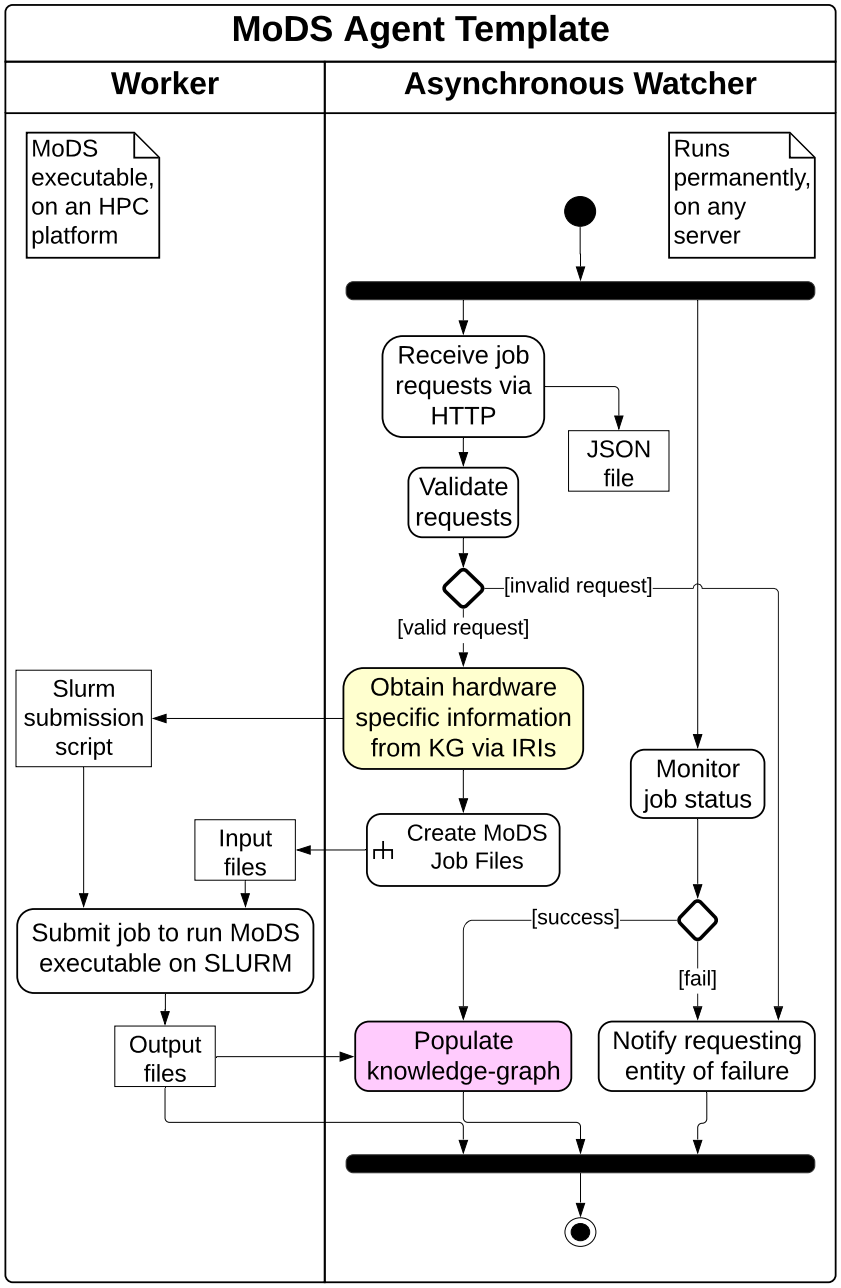


Fig. 5.3 UML activity diagram of a templating agent that enables MoDS jobs to be executed asynchronously on an HPC platform upon HTTP requests. The same design is followed by all MoDS wrapper agents, distinguished by different activate nodes for jobs files creation. The yellow-shaded action indicates the data-retrieving operation of agents over the dKG, whereas magenta refers to the dKG populating operation.

*k*inetics [56] for ignition delay time simulation and Cantera [131] for laminar flame speed simulation.

Figure 5.4 illustrates the automated process of generating MoDS job files. This process is structured in three layers: the file management centre communicates the input and output, a marshaller collects all information required by the MoDS executable, and finally the layer manages the individual executable models. In the file management centre, the process starts by querying the dKG for information related to each experiment response. This information is then passed to the marshaller to allocate executable models for simulating each response. The simulation files and execution script required for the selected models are then generated in the executable model layer and sent back to the marshaller.

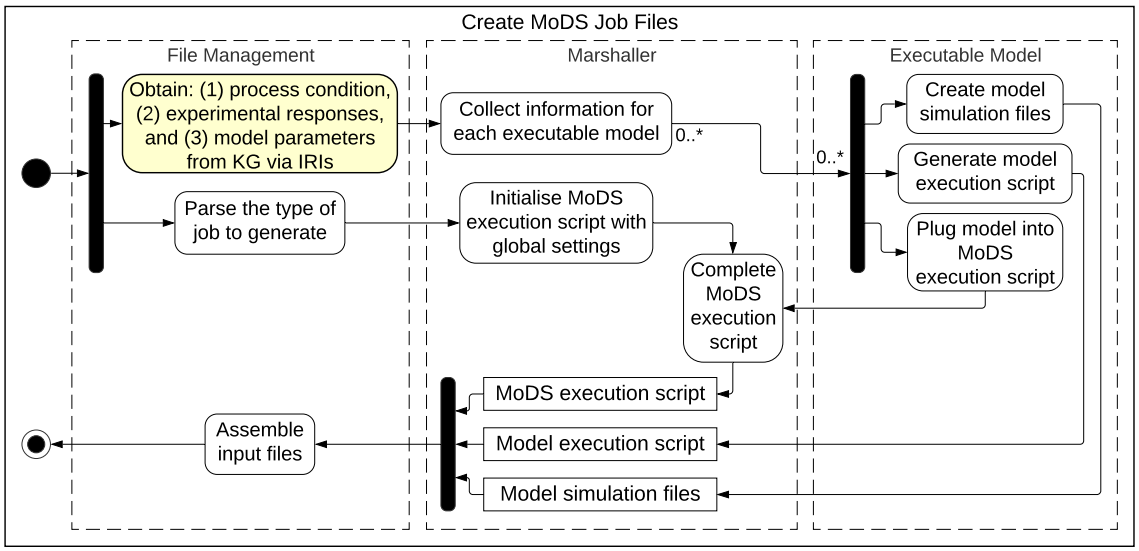


Fig. 5.4 Workflow of the process of creating requested job files. The whole process corresponds to the activity node in Fig. 5.3.

At the same time, the type of job requested is parsed in the file management centre and passed to the marshaller to initialise a MoDS execution script with predefined global settings. This script is connected to the selected models generated by the executable model layer. All generated files are then assembled in the file management centre and transferred to an HPC platform.

The first two stages in the automated mechanism calibration are performed by two MoDS template agents: MoDSSensAna Agent for sensitivity analysis and MoDSMechCalib Agent for mechanism calibration. Three parameters are currently available for the sensitivity analysis: the magnitude of the relative perturbation, the type of overall sensitivity (maximum absolute value or average absolute value), and the number of reactions to be optimised.

For the mechanism calibration, the parameters are made available in two folds: the global settings for the algorithms used in the MoDS job and the calibration objective parameters. For the algorithms, the total number of Sobol points to be generated and the termination tolerance of the Hooke-Jeeves algorithm can be specified by the user. For the calibration objective parameters, two parameters are available: the weighting in the objective function and the response scaling type (logarithmic or linear). Provision is also made for users to supply their own list of reactions to guarantee their inclusion in the calibration process.

A coordination agent manages the interactions between the MoDSSensAna Agent and MoDSMechCalib Agent with the dKG. These three agents form the overall automated mechanism calibration agent, AutoMechCalib Agent.

Figure 5.5 illustrates the UML sequence diagram of the AutoMechCalib Agent as a five-step process. Initially, the coordination agent validates the job request and invokes the MoDSSensAna Agent for a sensitivity analysis via IRIs. Secondly, the MoDSSensAna Agent communicates with the dKG via IRIs to obtain the chemical model to be calibrated and the process conditions over which the sensitivity analysis is to be conducted. After the sensitivity analysis, the MoDSSensAna Agent returns the list of IRIs of the identified reactions to the coordination agent. Thirdly, the coordination agent requests MoDSMechCalib Agent for a mechanism calibration job with the reactions identified. Fourthly, the MoDSMechCalib Agent carries out the calibration. Global search and local optimisation are used with experimental data retrieved from the dKG. As the final step, the MoDSMechCalib Agent populates the dKG with the calibrated mechanism and returns its IRI to the coordination agent. Figure 5.6 represents the derivation subgraph of such a coupled sensitivity analysis and mechanism calibration workflow.

The process relied upon linked ontologies. These provided the connection between combustion experiments (OntoChemExp) and kinetic mechanisms (OntoKin) via the unique identification of chemical species (OntoSpecies).

5.3 Results and discussions

5.3.1 Sensitivity study

Prior to the automated calibration task, a sensitivity study was performed to assess the effect of the relative perturbation size used during the reaction selection sensitivity analysis in the automated calibration process. This sensitivity study was performed using the MoDSSensAna Agent. The study involved 21 relative perturbation sizes for the finite difference approximations (r in Eq. 5.3) of the derivatives required for the sensitivity coefficients

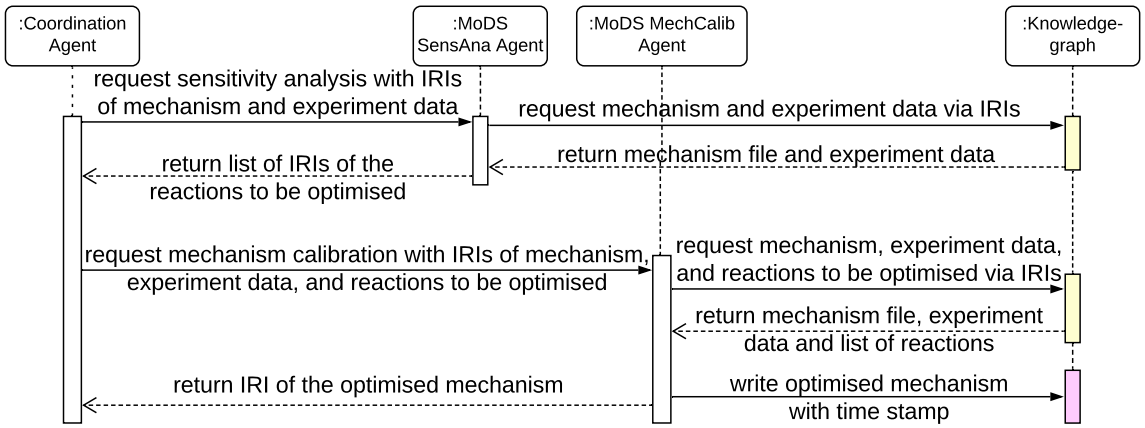


Fig. 5.5 UML sequence diagram of the automated mechanism calibration process that captures the interaction between the different agents and the dKG. Actions where the agent retrieves data from the dKG are shaded in yellow and those where the agent populates the dKG in magenta.

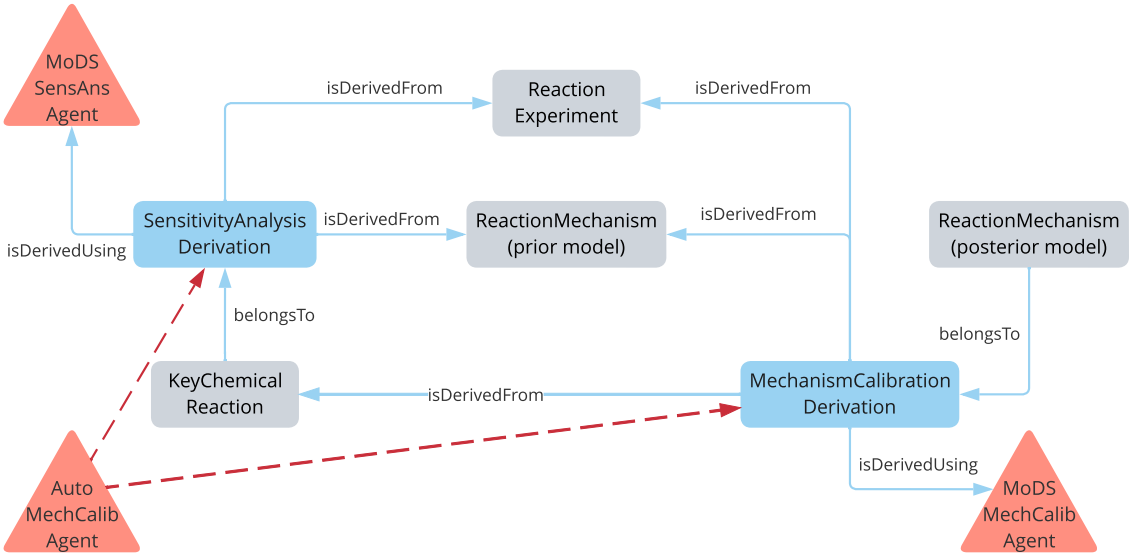


Fig. 5.6 The information flow expressed in the derived information framework during the sensitivity analysis and mechanism calibration workflow. The red dashed lines refer to instantiation by the agent.

$(1 \times 10^{-n}, 2 \times 10^{-n}$, and 5×10^{-n}; $n = 1, \ldots, 7)$ and assess the sensitivities for both ignition delay times and laminar flame speeds to the Arrhenius pre-exponential factor for all 215 reactions. For every reaction, a normalised sensitivity coefficient was computed for all 73 experimental conditions used for the calibration process. The maximum absolute value form of the sensitivity coefficient was used and the reactions were ranked based on this value to

determine their relative importance. We present the 10 reactions with the greatest absolute sensitivity coefficients.

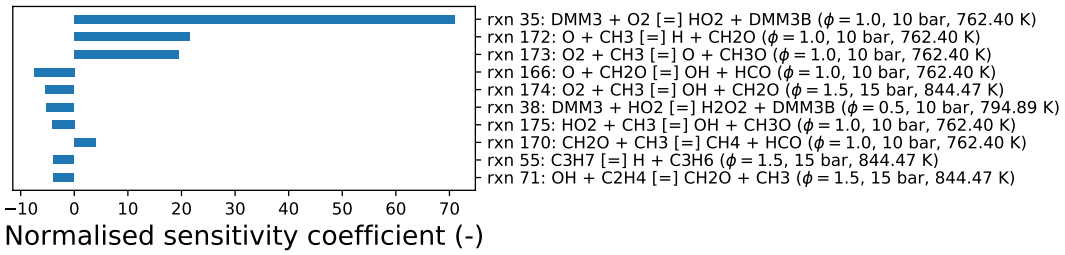
Figure 5.7 presents the ignition delay time results of the sensitivity study. The values of the sensitivity coefficients at their maximum absolute value are shown in Fig. 5.7(a) along with the conditions for each case. The effect of temperature on the sensitivities at fixed equivalence ratio and pressure is shown in Fig. 5.7(b). As demonstrated by the variability in normalised sensitivity coefficients to changing conditions, different sets of active parameters can be identified. This includes in particular different active parameters for low *vs.* high temperature regions. It is thus necessary to assess reaction importance through a *global* perspective. Figure 5.7(c) shows the stability of the sensitivity coefficients over the range of relative perturbation sizes investigated. The horizontal dashed lines bound the region that clearly illustrates the variation for all reactions, as magnified and presented in Fig. 5.7(d). It can be seen that the model, like most combustion models, is highly non-linear. The selection of suitable relative perturbation sizes is thus in favour of small values where the model behaves in a relatively linear fashion for most of the reactions. The peaks in Fig. 5.7(b) correspond to the sensitivity coefficients used for ranking the reactions, matching the values at the vertical dotted line in Fig. 5.7(d) and those presented in Fig. 5.7(a).

This study identified two reactions from the $PODE_3$ sub-mechanism, reactions 35 and 38. Reaction 35 belongs to the first O_2 addition reaction class. This class is suggested by Ren et al. [294] as a good choice for calibration as ignition delay times are usually sensitive to this class of reactions at low initial temperatures. Reaction 38 involves H-abstraction by HO_2 radicals, shown to increase the fuel reactivity thus important to $PODE_3$ combustion [38].

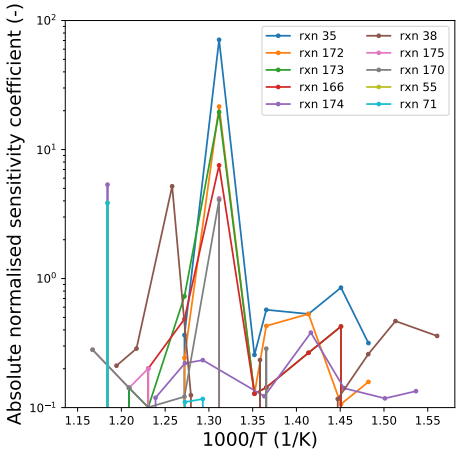
Figure 5.8 presents the sensitivity analysis results for laminar flame speeds. The structure remains similar to that of Fig. 5.7 but with Fig. 5.8(b) showing the effect of the equivalence ratio rather than temperature.

Reaction 122 was found to have the greatest influence on the laminar flame speed. This is a chain-branching step $O_2 + H \longleftrightarrow OH + O$ and is expected to have a significant effect on the laminar flame speed [377, chap. 8]. Other reactions identified mostly involve small species and radicals which governs a large part of the heat release (*e.g.*, $CO + OH \longleftrightarrow CO_2 + H$).

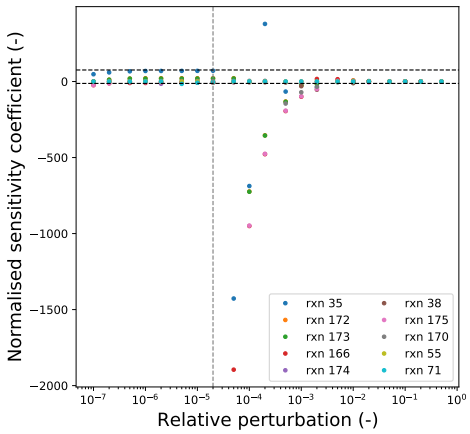
The reactions identified for both laminar flame speed and ignition delay times fall in line with those selected while Lin et al. [213] constructing the reduced mechanism. The mechanism was developed using a decoupling methodology, separating the mechanism detail into three levels: detailed for $H_2/CO/C_1$, reduced for $C_2{-}C_3$, and skeletal for $C_4{-}C_N$. As appeared in the list of reactions found to be sensitive for laminar flame speed, most of them are in the detailed $H_2/CO/C_1$ sub-mechanism, representing high temperature combustion



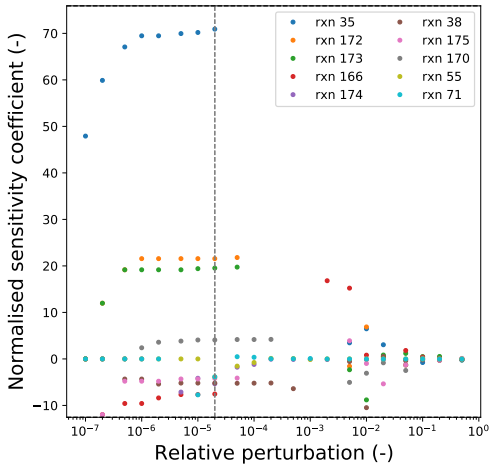
(a) Selected 10 most sensitive reactions using a relative perturbation of 2×10^{-5}, the labelled condition for each reaction corresponds to that at which the peak value in Fig. 5.7(b) was obtained.



(b) Sensitivities as function of temperature.

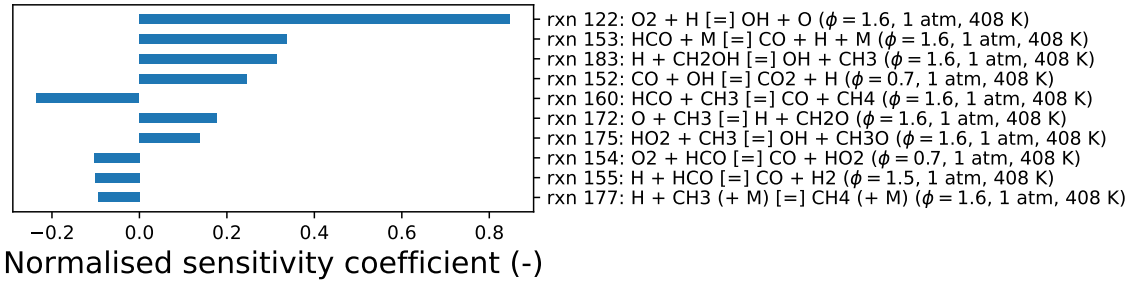


(c) Sensitivities as function of relative perturbation.

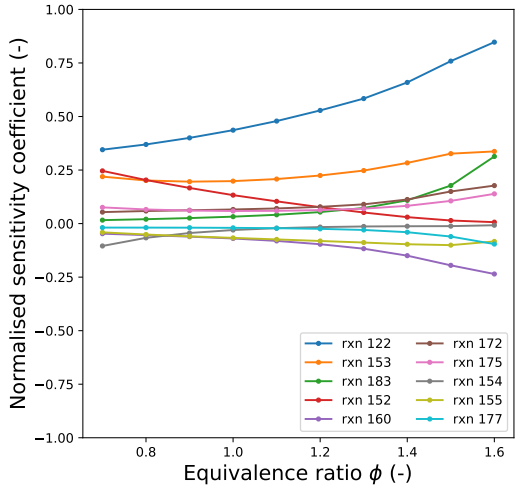


(d) Same as Fig. 5.7(c), but for a smaller range of sensitivities as indicated by the horizontal lines.

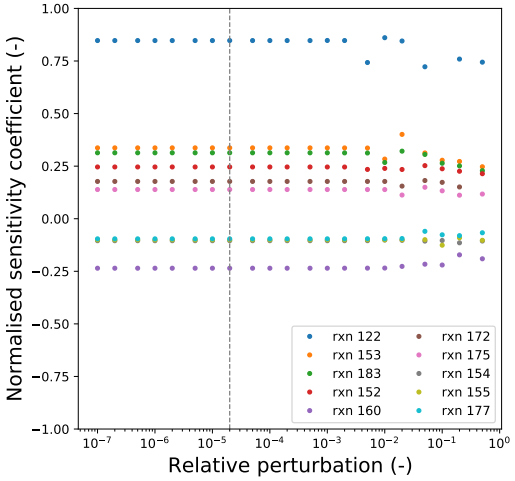
Fig. 5.7 Sensitivity analysis of the ignition delay times with respect to Arrhenius pre-exponential factors in the starting mechanism. The list of reactions is selected based on the maximum value of its sensitivities across all considered points in experimental condition space with a relative perturbation of 2×10^{-5}.



(a) Selected 10 most sensitive reactions using a relative perturbation of 2×10^{-5}.



(b) Sensitivities as function of equivalence ratio.



(c) Sensitivities as function of relative perturbation.

Fig. 5.8 Sensitivity analysis of the laminar flame speed with respect to Arrhenius pre-exponential factors in the starting mechanism. The list of reactions is selected based on the maximum value of its sensitivities across all considered points in experimental condition space with a relative perturbation of 2×10^{-5}.

process. While for ignition delay times, the most sensitive reaction is from a skeletal structure for C_4-C_N, representing low temperature combustion process.

Following this analysis, a relative perturbation size of 2×10^{-5} was chosen for the calibration process. This value is selected to act as a trade-off between numerical errors and the onset of non-linearities, given that larger sizes present systematic changes in sensitivity as a function of perturbation, specifically for ignition delay times, whereas smaller values increase the likelihood of problems due to rounding errors.

5.3.2 Mechanism calibration

In parameter estimation tasks, deciding the number of parameters to include within the estimation task requires consideration of the trade-offs between precision and tractability. For this calibration case, the reactions to optimise have been set as the top 10 reactions identified for ignition delay times and laminar flame speeds by the MoDSSensAna Agent. This resulted in a total of 18 reactions to optimise due to reactions appearing in both sensitivity analyses.

Another trade-off requiring consideration is that of the weighting between ignition delay times and laminar flame speeds. In many cases, differing quantities of experimental data are available and there may exist differences in users' preference in weightings. The weighting of the two is handled by the value of α in the objective function of the calibration process (*i.e.*, Eq. 5.6). In this case, there are 63 data points on the ignition delay time and 10 on the laminar flame speed. Correcting this imbalance in the number of experimental results forms a natural starting point for selecting a value for α, and so values of α in the range of 6.3 to 1 were investigated. This range is intended to cover values that offer a good balance between the two responses and prevent the domination of ignition delay times for the calibration.

The calibration routine seeks to optimise the values of the pre-exponential factors for the target reactions. The range selected for the task was 10^{-2} to 10^{2} times the original value. This is a relatively large range in relation to physical uncertainties, however, this is a reduced mechanism and individual reactions must not be misinterpreted as elementary, physical reactions. The process began with Sobol sampling within the selected range of values. 10^{4} logarithmic-evenly distributed points were used to determine three starting points for the optimisation routine that displayed the lowest values of the objective function.

Following the sampling stage, a Hooke-Jeeves optimisation routine is performed. The routine was performed with 400 iterations and a termination step size of 0.001, with an initial step size of 0.2 and a step size reduction factor of 0.5. The results of the sampling and optimisation stages are presented in Table 5.2.

Prior to the sampling stage, the scaled sum-of-squares-errors value was found to be 14554 for ignition delay times and 774 for laminar flame speeds, resulting in objective function values of 19430 and 15328 for α values of 6.3 and 1, respectively. This indicates the value of performing the initial Sobol sampling stage, with a significant improvement in the objective function being achieved prior to any optimisation. This is particularly valuable as the Hooke-Jeeves algorithm performs a local search, significantly benefiting from a good initial point.

The optimisation stage is further seen to be providing an improvement in the objective function, significantly reducing its value from the sampling stage. The results of the optimi-

Table 5.2 Objective function of global search and local optimisation results of the starting mechanism. The best three Sobol points from global search, *i.e.*, the three Sobol points with the smallest objective function values, were chosen for further local optimisation with a Hooke-Jeeves (H-J) algorithm. Each of these Sobol points represents a combination of the active parameters sampled within the selected range. Values in boldface indicate the best performing mechanism for each response ratio and are chosen as the starting mechanism for the next iteration of calibration.

Ratio α	Best Sobol	H-J	2^{nd} Sobol	H-J	3^{rd} Sobol	H-J
6.3	4446	**659**	5153	1375	5356	1263
4.98	4170	**657**	4697	1378	4788	895
3.65	3892	**710**	4216	712	4238	1398
2.33	3617	896	3648	**714**	3783	1373
1	3075	654	3165	**653**	3324	1370

sation stage are comparable, in the sense of having the same order of magnitude, as those of Lin et al. [213], which achieved an objective function value of 140 with an α value of 1.

Although variation is seen in the objective function values after the sampling stage in response to changing α values, this same change is not observed after the optimisation stage. This is a result of the contributions to the objective function from the laminar flame speeds becoming very small after optimisation. The laminar flame speed is largely governed by small molecule oxidation which remains a detailed sub-mechanism within the decoupling methodology adopted in the Lin et al. [213] mechanism. This suggests a better fit would be expected for both the initial mechanism and calibrated mechanism for laminar flame speeds than ignition delay times.

During the initial sensitivity analysis, the mechanism is unable to accurately reproduce the combustion characteristics. This suggests that reaction selection at this stage may be premature and may not select all reactions that are of the most importance local to the optimum fitting. Additional reactions may also only become important after the rates of the initially identified reactions are closer to their optimum values. For these reasons, a further iteration of the calibration algorithm is necessary, which is consistent with the recommendation by Frenklach [109].

5.3.3 Second iteration

A second calibration was performed on the best-performing mechanism for each value of α. The ontological structure of the framework aided in this process, allowing for the task to be completed by the passing of the IRIs for the experiments and calibrated mechanisms from the

last iteration to the AutoMechCalib Agent agent with the rest of the configuration identical to the first iteration.

After validating the job request, the coordination agent requested the MoDSSensAna Agent to perform a sensitivity analysis to identify key reactions. Since the sensitivities depend not only on the conditions, but also on the model parameters, the active parameters identified are different. The list of IRIs for the updated active parameters was then added to the original job request by the coordination agent and passed to the MoDSMechCalib Agent. A mechanism calibration was then performed to optimise the mechanism. All other settings for the sensitivity analysis (*i.e.*, relative perturbation size, the type of overall sensitivity, and the number of reactions to be optimised), as well as mechanism calibration (*i.e.*, the global settings for the algorithms and the calibration objective parameters) were left unchanged. The results after both the sampling and calibration stages are summarised in Table 5.3.

Table 5.3 Objective function values after sampling and optimisation for the best-performing mechanisms selected from the first iteration of mechanism calibration for each α value. All mechanisms showed significant improvement in this iteration of calibration, with the best-performing mechanism underlined.

Ratio α	Best Sobol	H-J	2^{nd} Sobol	H-J	3^{rd} Sobol	H-J
6.3	605	278	627	296	651	**260**
4.98	375	89	505	**79**	507	187
3.65	459	**243**	500	259	552	253
2.33	571	**38**	658	133	702	127
1	355	151	376	**133**	377	156

After the calibration stage, the best-performing mechanism was found with an α value of 2.33. The objective value of the calibrated mechanism ($\Phi = 38$) is found to show a 79% decrease compared to that of the Lin et al. [213] mechanism ($\Phi = 181$) when the same α values of 2.33 are used in the current definition of the objective function (Eq. 5.6).

The performance of the mechanism of Lin et al. [213] (manual calibration) and the mechanism of this work (automated calibration) is compared in Fig. 5.9 and Fig. 5.10. The automatically calibrated mechanism shows a good fit to the experimental data in all cases. It should be noted that in the original paper of Lin et al. [213], a temperature rise of 400 K criterion was used for calibrating and assessing their model against ignition delay times, yielding an objective function value of 121 according to Eq. 5.6. In this work, a maximum rate of pressure increase criterion is used when comparing the models' performance in Fig. 5.9, resulting in an objective function value of 108 for the Lin et al. [213] model. This change of ignition criterion brings it in line with that used for the experimental results, and we note that this represents an improvement for the Lin et al. [213] model.

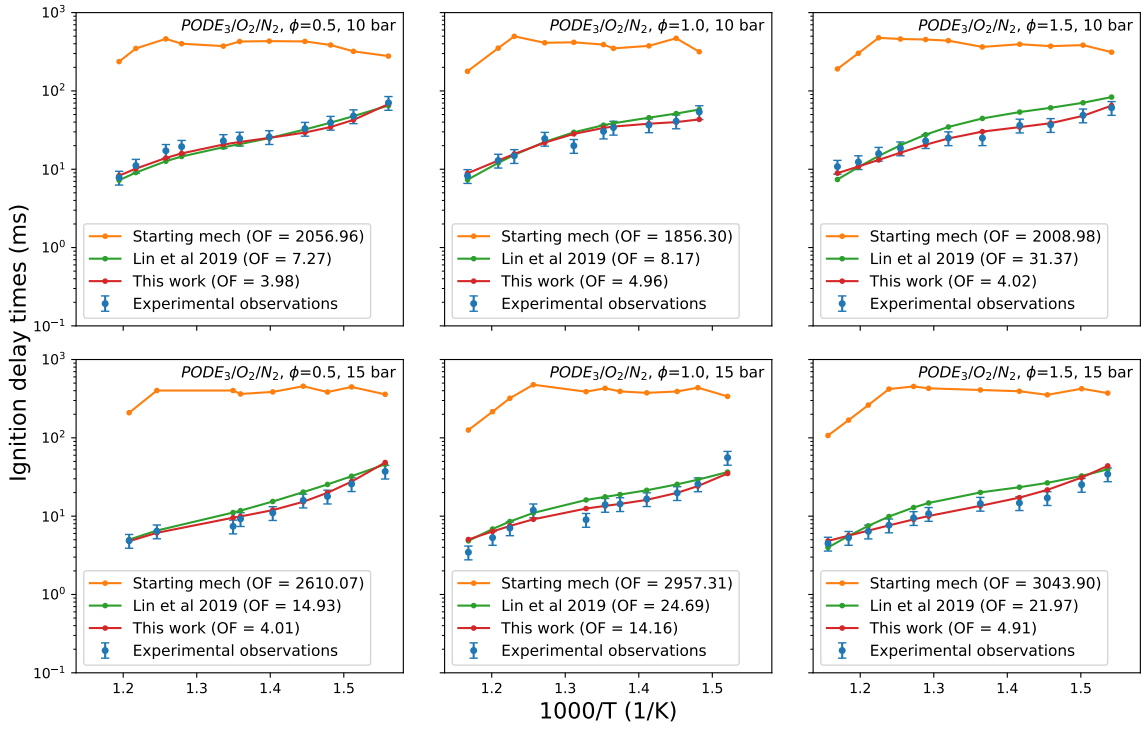


Fig. 5.9 Comparison of the mechanisms from [213] and the AutoMechCalib Agent agent (this work) at simulating ignition delay times (maximum rate of pressure increase ignition criterion) of $PODE_3/O_2/N_2$ mixtures at three equivalence ratios [151]. The model performance is displayed as the ignition delay time contribution to the objective function. As per the experimental results, the oxidizer used in this study has different compositions: (1) $\phi = 0.5$, O_2:$N_2 = 1:8$; (2) $\phi = 1.0$, O_2:$N_2 = 1:15$; (3) $\phi = 1.5$, O_2:$N_2 = 1:20$.

The Negative Temperature Coefficient (NTC) behaviour of the fuel is captured in both the mechanism of this work and that of Lin et al. [213]. In Lin et al. [213], it is claimed that capturing the NTC region is achieved through optimisation of the isomerisation reaction $DMM_3BO_2 \longleftrightarrow DMM_3OOH_{35}$. In this work, this reaction was not identified as important and so was not calibrated, instead remaining at the same value as used in He et al. [151] where no apparent NTC region was captured.

It is believed the capturing of the NTC behaviour in this work is a result of the sensitivity analysis identifying reactions of importance in the intermediate-temperature regime (around 770 K), corresponding to the NTC region. This effect may be seen in Fig. 5.7(b) where the majority of the sensitivities show a peak in the intermediate-temperature region.

Table 5.4 summarises the changes made to the Arrhenius pre-exponential factors during the calibration process. The range of adjustment for the rate parameters during the calibration process was 10^{-4} to 10^{4}. Whilst this may be considered a wide range, we note that other studies dealing with reduced mechanisms report similar orders of adjustment, such as Lin

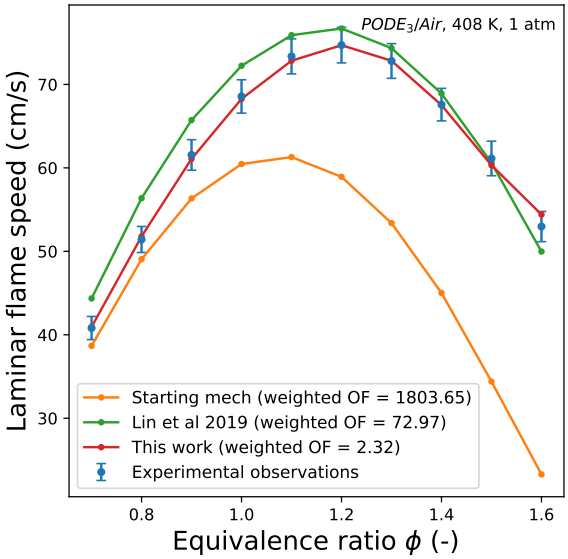


Fig. 5.10 Comparison between the model from [213] and the AutoMechCalib Agent agent (this work) on simulating laminar flame speed of $PODE_3$/Air mixtures at atmospheric pressure and an initial temperature of 408 K [348]. Model performance is displayed as the value of the laminar flame speed contribution to the objective function with an α value of 2.33.

et al. [213] and Chang et al. [46] while calibrating mechanisms constructed with decoupling methodologies. In contrast to Lin et al. [213] and Chang et al. [46] however, the reduced mechanism in this work is optimised as a whole to fit the provided experimental data. It is further noted that even more complete $PODE_3$ mechanisms, such as that of Ren et al. [294], modify the pre-exponential factors by an order of magnitude during calibration. This is to balance necessary levels of adjustment against unnecessarily large search spaces.

Two additional H-abstraction reactions from the $PODE_3$ sub-mechanism and a total of eight reactions were identified by the second calibration iteration that were not identified by the first. The second iteration fully captures the governing reactions of the low-temperature combustion process, as found to be important to modelling ignition delay times [213]. Thus, the substantial improvement in model performance found in the second iteration is not surprising.

Having been optimised in two stages against 73 data points, using 18 and 19 active parameters respectively, the model is seen to agree well with the available data points, capturing major trends without incorporating noise present in the experimental data. Although the model performance is not assessed for process conditions outside the range used in this study, one of the aims of the dKG approach is that the calibration can be easily repeated once new data are made available.

Table 5.4 Summary of the calibrated Arrhenius pre-exponential factors. Omitted values imply a reaction rate is unchanged. The unit of the pre-exponential factors is $m^3mol^{-1}s^{-1}$ or s^{-1} for two and one reactant respectively. The indexes of reactions follow the labels generated while converting the mechanism from CHEMKIN to OntoKin format. Reactions identified as sensitive for different responses are denoted as † for ignition delay time and ‡ for laminar flame speed. Note that $PODE_3$ is denoted as DMM_3.

Reaction	Equation	Original A factor	1^{st} iteration	2^{nd} iteration
35	$DMM_3 + O_2 \leftrightarrow HO_2 + DMM_3B$	6.66×10^6	6.66×10^8†	1.37×10^{10}†
36	$DMM_3 + OH \leftrightarrow H_2O + DMM_3B$	3.79×10^{-2}	-	4.22×10^{-4}†
37	$DMM_3 + H \rightarrow H_2 + DMM_3B$	7.40×10^6	-	5.05×10^5†
38	$DMM_3 + HO_2 \leftrightarrow H_2O_2 + DMM_3B$	4.00×10^7	2.25×10^9†	1.24×10^9†
55	$C_3H_7 \leftrightarrow H + C_3H_6$	1.25×10^{14}	1.08×10^{13}†	-
71	$OH + C_2H_4 \leftrightarrow CH_2O + CH_3$	1.00×10^8	3.23×10^6†	-
122	$O_2 + H \leftrightarrow OH + O$	1.04×10^8	1.04×10^{10}‡	1.72×10^{11}‡
124	$H_2 + O \leftrightarrow OH + H$	8.79×10^8	-	1.35×10^8‡
126	$OH \leftrightarrow H_2O + O$	3.34×10^{-2}	-	4.19×10^{-3}‡
152	$CO + OH \leftrightarrow CO_2 + H$	2.23×10^{-1}	6.22×10^{-1}‡	1.16×10^1‡
153	$HCO + M \leftrightarrow CO + H + M$	5.75×10^5	2.13×10^5‡	8.30×10^5‡
154	$O_2 + HCO \leftrightarrow CO + HO_2$	7.58×10^6	2.02×10^6‡	-
155	$H + HCO \leftrightarrow CO + H_2$	7.23×10^7	4.63×10^9‡	1.04×10^8‡
160	$HCO + CH_3 \leftrightarrow CO + CH_4$	1.20×10^8	9.40×10^9‡	3.80×10^{10} †‡
166	$O + CH_2O \leftrightarrow OH + HCO$	1.81×10^7	1.21×10^7†	-
167	$OH + CH_2O \leftrightarrow H_2O + HCO$	3.43×10^3	-	6.79×10^3†
170	$CH_2O + CH_3 \leftrightarrow CH_4 + HCO$	3.64×10^{-12}	5.78×10^{-11}†	5.38×10^{-9}†
172	$O + CH_3 \leftrightarrow H + CH_2O$	8.43×10^7	8.43×10^5 †‡	-
173	$O_2 + CH_3 \leftrightarrow O + CH_3O$	1.99×10^{12}	1.99×10^{14}†	7.25×10^{14}†
174	$O_2 + CH_3 \leftrightarrow OH + CH_2O$	3.74×10^5	6.56×10^6†	9.33×10^4†
175	$HO_2 + CH_3 \leftrightarrow OH + CH_3O$	1.00×10^6	1.00×10^8 †‡	-
177	$H + CH_3 (+ M) \leftrightarrow CH_4 (+ M)$	1.27×10^{10}	1.03×10^9‡	-
180	$OH + CH_4 \leftrightarrow H_2O + CH_3$	5.72×10^0	-	5.03×10^2‡
183	$H + CH_2OH \leftrightarrow OH + CH_3$	9.64×10^7	3.30×10^9‡	3.78×10^9‡
186	$O_2 + CH_2OH \leftrightarrow HO_2 + CH_2O$	2.41×10^8	-	2.41×10^{10}‡
193	$CH_3O + M \leftrightarrow H + CH_2O + M$	8.30×10^{11}	-	2.26×10^{12}†

The calibrated mechanism is publicly available at doi:10.17863/CAM.59826. It should be noted that the chemical model should only be used as a whole and individual rate parameters should not be used outside of this model. This particularly applies to reactions whose rates are well-established in the literature, with relatively narrow uncertainty bounds. For such reactions, the adjusted rates as part of a reduced and calibrated mechanism such as the one in this work may well be unphysical in the sense that they have been adjusted well beyond their established uncertainty bounds. One approach to alleviate this problem, as taken by Lin et al. [213], is to calibrate only those reactions whose rates are not well-known (here the $PODE_3$ sub-mechanism), whilst leaving the ones with well-known rates unchanged (*e.g.*, the core C_1-C_3 chemistry). A more general approach is to calibrate the chosen reactions

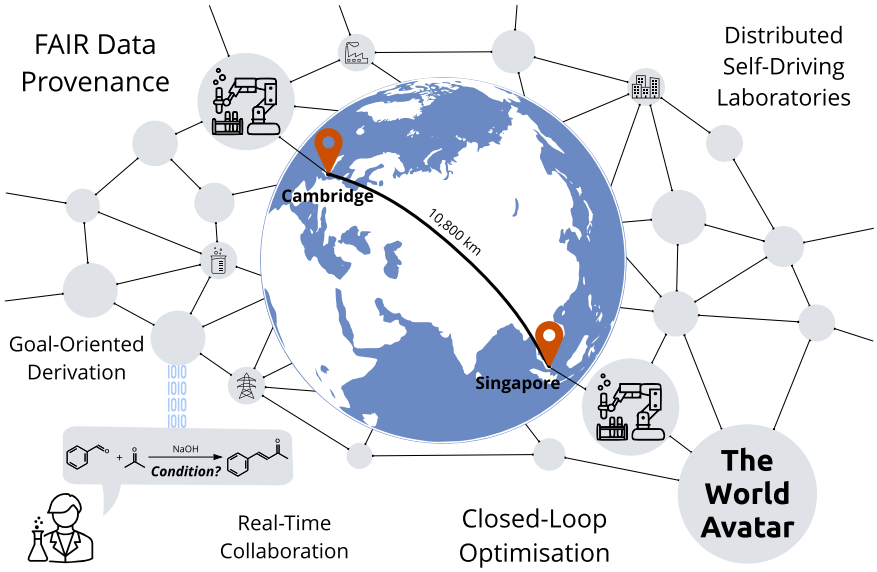
within their respective ranges of uncertainty (see *e.g.*, Sheen and Wang [328]). As the focus of the present work is to take the first steps in the proof-of-concept of the dKG approach, we have omitted such treatments for simplicity at this stage. Taking uncertainties into account is, however, a natural next step, and this should be addressed in future work.

5.4 Chapter summary

In this chapter, we applied the dKG approach to automate mechanism calibration. The ontology facilitated translation between various community simulation input file formats and agents standardised calculation tasks for different experimental measurements. This combination represents a step towards greater provenance determination for combustion mechanisms by linking models to the experimental results used in their calibration. As a demonstration of these technologies, a case study was used of a reduced $PODE_3$ combustion mechanism. Two iterations of the coupled agent process were required to sufficiently optimise the initially poor-fitting mechanism, where users only needed to provide the IRIs of experimental data and the mechanism to be calibrated. As executed on an HPC facility, this chapter demonstrated the effectiveness of dKG in developing a computational model that fits the data significantly more accurately as measured by the stated objective function in a short time span.

Chapter 6

Automated experiment execution and optimisation



"We choose to go to the moon in this decade and do the other things, not because they are easy, but because they are hard, because that goal will serve to organise and measure the best of our energies and skills, because that challenge is one that we are willing to accept, one we are unwilling to postpone, and one which we intend to win, and the others, too."

– John F. Kennedy, Address at Rice University on the Nation's Space Effort (1962)

The chapter draws from two papers: one published in *Nature Communications* and another currently *in preparation* in collaboration with the Computational Modelling Group and Sustainable Reaction Engineering at the University of Cambridge, CMCL Innovations, and Cambridge CARES. Dr Taylor and Dr Karan developed the experiments in Cambridge and Singapore, respectively, for the distributed SDLs use case. Dr Taylor developed the experiments in the Suzuki coupling use case. Dr Mosbach and Dr Lee assisted in the design of ontologies and agents. All authors provided feedback. The ontology/agent development, deployment, analysis and writing were performed by the author.

In this chapter, the dKG approach is expanded to automate wet-lab experiments. This approach utilises ontologies to capture the data and material flows involved in DMTA cycles, and employs autonomous agents as executable knowledge components to carry out the experimentation workflow. All data provenance is recorded following the FAIR principles, ensuring its accessibility and interoperability. The approach is demonstrated in two practical applications. The first use case is distributed SDLs, where two robots in Cambridge and Singapore are linked to achieving collaborative closed-loop optimisation in real-time. The two robots effectively produced a Pareto front for the cost-yield optimisation problem over the course of three days of operation. The second use case involves catalyst selection for a multi-step Suzuki coupling reaction where the dKG approach achieved efficient synthesis of challenging molecules. In both applications, the dKG evolves autonomously while progressing towards the research goals set by the scientist.

6.1 Introduction

The concept of laboratory automation, recently reinterpreted as SDLs [146, 1], has been in existence since the 1960s, when Merrifield et al. [233] introduced the first automated chemistry hardware. Since then, SDLs have gained widespread adoption in chemistry [59, 347, 35, 50], materials science [288, 350, 403], biotechnology [190, 91, 34] and robotics [392], resulting in accelerated scientific discovery and societal development. However, the implementation of SDLs can be challenging and typically requires a highly specialised team of researchers with expertise in chemistry, engineering, and computer science. Consequently, studies are often conducted by large research groups within a single organisation. Even in cases where collaborations occur between research groups, the SDL is usually centralised within the same laboratory.

In response to the pressing global challenges of today, there is a growing consensus within the scientific community that a paradigm shift towards a globally collaborative research

network is necessary [342, 77, 207]. Such a connected network holds great potential in supporting various tasks ranging from automating the characterisation of epistemic uncertainty in experimental research [295] to advancing human exploration in deep space [311]. This shift requires decentralising SDLs to integrate different research groups to contribute their expertise towards solving emerging problems [391, 326, 346]. As discussed in Chapter 1, achieving this vision is not an easy task and entails four major challenges. Many attempts have been made to tackle these challenges with different focuses. For resource orchestration, middleware such as ChemOS [302, 333], ESCALATE [277], and HELAO [289] exist to glue different components within an SDL and abstract the hardware resources. For data sharing, χDL [229, 304, 152] and AnIML [313] are examples of standard protocols developed for synthesis and analysis respectively. In the realm of data provenance, Mitchell et al. [237] proposed a data pipeline to support the modelling of the COVID pandemic, whereas Statt et al. [344] devised a KG to record experimental provenance in materials research. Although these studies provide insights into building a collaborative research environment and many of them are open-sourced, they are developed in isolation with customised data interfaces. Enhancing interoperability both within and between these systems is essential to establish a truly connected research network.

As demonstrated in the previous chapters, semantic web technologies such as KGs [161] offer a viable path forward. Ontologies abstract both resources and data using the same notion to encompass all aspects of scientific research laboratories: The experiment itself, including its physical setup and underlying chemistry; moving handlers that can be of human or robotic nature; and the laboratory providing necessary infrastructure and resources. Consequently, this allows for a common language between participants when allocating tasks and sharing results. Going beyond static knowledge representation, we can encode software agents as executable knowledge components to enable dynamicity and continuous incorporation of new concepts and data while preserving connections to existing information. As the dKG expands, this characteristic allows for capturing data provenance from experimental processes as knowledge statements, effectively acting as a living copy of the real world. This dKG streamlines the immediate dissemination of data between SDLs, offering a promising holistic solution to the aforementioned challenges and the pursuit of the Nobel Turing Challenge [191, 188, 298].

The purpose of this chapter is to demonstrate a proof-of-concept for automated experiments enabled by the dKG approach. This work forms the foundation of a holistic approach to lab automation by including diverse aspects of research laboratories in an all-encompassing digital twin. By employing dKGs that integrate knowledge models from different domains, we can address the challenges related to interoperability and adaptability

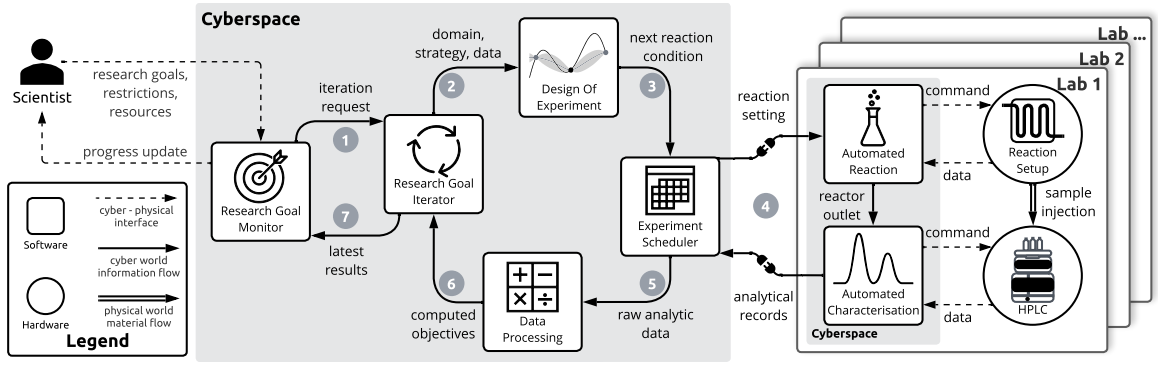
commonly encountered in the platform-based approach as reviewed in Chapter 3. The goal-driven architecture facilitates reasoning across the knowledge base, allowing high-level, abstract goals to be decomposed into specific sub-goals and more tangible tasks. Within this framework, humans play a dual role, functioning both as goal setters and operators (when necessary) for executing and intervening in experiments. When acting as operators, humans can be represented in the KG similarly to robots, and they receive instructions in a human-readable format. This facilitates the realisation of a hybrid and evolving digital laboratory, bridging potential "interim technology gaps" [162]. To illustrate the effectiveness of this approach, we present a demonstration using two robots in Cambridge and Singapore collaborating on a multi-objective closed-loop optimisation problem in response to a goal request from scientists. The framework is also demonstrated on a Suzuki coupling use case for a multi-step Suzuki coupling reaction. The operations described in this chapter are carried out through robotic handling, with humans primarily involved in the preparation of initial materials and the maintenance of the equipment.

6.2 Methodology

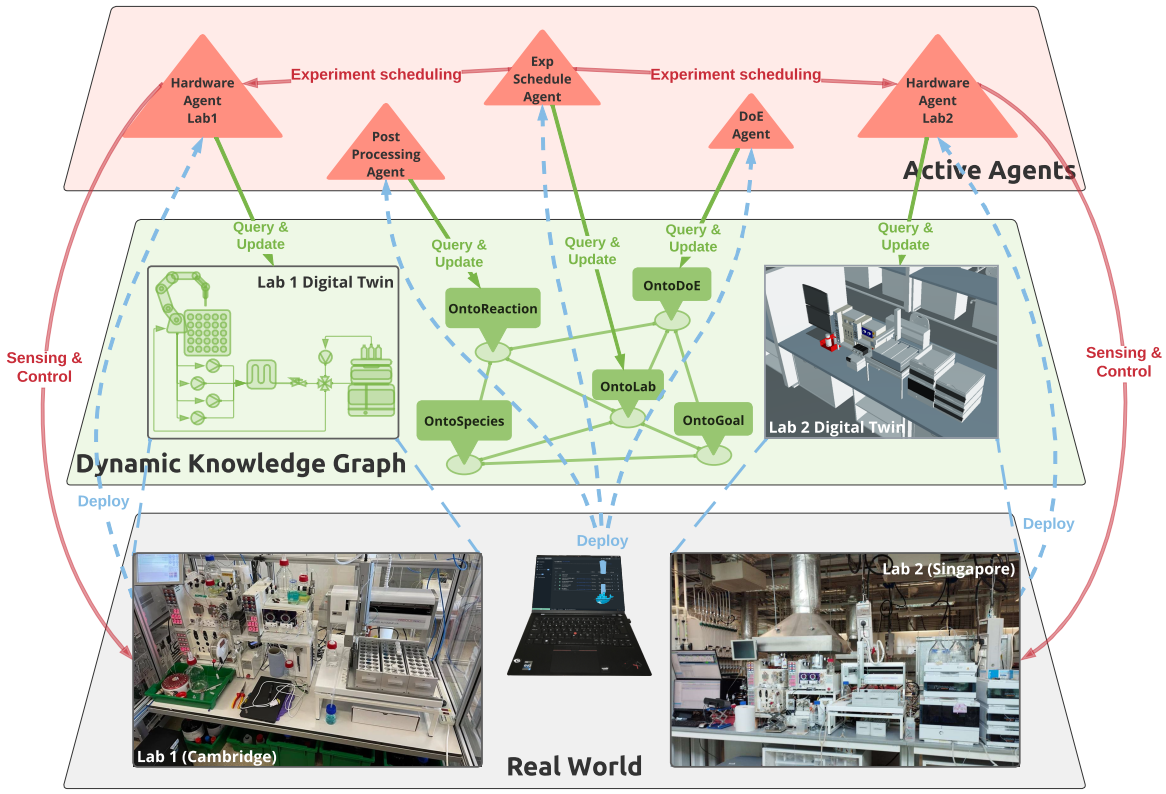
6.2.1 Architecture of distributed SDLs

Closed-loop optimisation in SDLs is a dynamic process that revolves around DMTA cycles [60, 354]. Compared to machine learning systems and scientific workflows that only capture data flows, SDLs offer an integrated approach by orchestrating both computational and physical resources. This involves the integration of data and material flows, as well as the interface that bridges the gap between the virtual and physical worlds. To this end, we propose a conceptual architecture of distributed SDLs that effectively incorporates all three flows, as illustrated in Fig. 6.1(a).

The proposed architecture presents a framework to enable scientists to set research goals and resource restrictions for a particular chemical reaction and have them trigger a closed-loop process in cyberspace. The process is initiated by the monitoring component, which parses the research goals and requests the iterations needed to achieve the objectives. The iterating component collects prior information about the design space and passes it on to the component that designs the next experiment. The algorithm employed [120, 55], as well as the availability of prior data, determines the combination of design variables to be proposed within the domain provided by the scientist. Similar to the scheduling of HPC jobs [393], the proposed physical experimentation is scheduled for execution in one of the available laboratories. The suggested conditions are translated to the machine-actionable recipe that



(a) Conceptual framework of components used to build a network of distributed SDLs for closed-loop optimisation. Information and materials from a chemical reaction flow autonomously across cyber and physical space until they accomplish the research goals set by scientists or the allocated resources have been consumed.



(b) A dKG approach. The overall system comprises three layers: the real world where the hardware is located, the dynamic KG that hosts all information in cyberspace, and the layer of active agents that manages the KG.

Fig. 6.1 An illustration of a distributed SDLs architecture.

enables the control of hardware for reaction and characterisation. In the physical world, this is reflected in the material flow between the two pieces of equipment. The data processing component is then responsible for computing the objectives by analysing the complete job information and raw data. If the resources are still available, a comparison of these objectives with the research goals determines whether the system should proceed to the next iteration.

This architecture liberates the scientists from routine work, however, it also poses challenges in the implementation in terms of ensuring robustness, scalability, maintainability, safety, and ethics. Ideally, the system should enable seamless integration of new devices, resources, and algorithms without disrupting the system's overall functioning. It is also critical to allow for dynamic adaption to changes in research goals and resource restrictions.

We believe dKG technology can help with realising this architecture. Specifically, as illustrated in Fig. 6.1(b), this technology abstracts the software components as agents that receive inputs and produce outputs. The flow of data between these components is represented as messages exchanged among these agents. Physical entities can be virtualised as digital twins in cyberspace, enabling real-time control and eliminating geospatial boundaries when multiple labs are involved. This reformulation of the closed-loop optimisation problem as information travelling through the dKG and reflecting their changes in the real world offers a powerful framework for achieving true distributed SDLs. In this way, we can think of an occurrence of physical experimentation as a sequence of actions that dynamically generates information about a reaction experiment as it progresses in time, analogous to computational workflows.

6.2.2 Chemical ontologies and digital twins

The realisation of SDLs requires a connection between abstract chemistry knowledge and concrete hardware for execution [382]. This calls for a set of connected ontologies, as identified in the analysis in Chapter 3 on the gaps in current semantic representations for chemical digitalisation. Figure 6.2 presents a selection of concepts and relationships as an effort to address these gaps. These concepts span various levels of abstraction involved in scientific research, ranging from the high-level research goals, through the conceptual level of chemical reactions and the mathematical level of design of experiments, down to the physical execution of reaction experiments and the laboratory digital twin. The modelling details of each ontology are described below with their cross-domain characteristics.

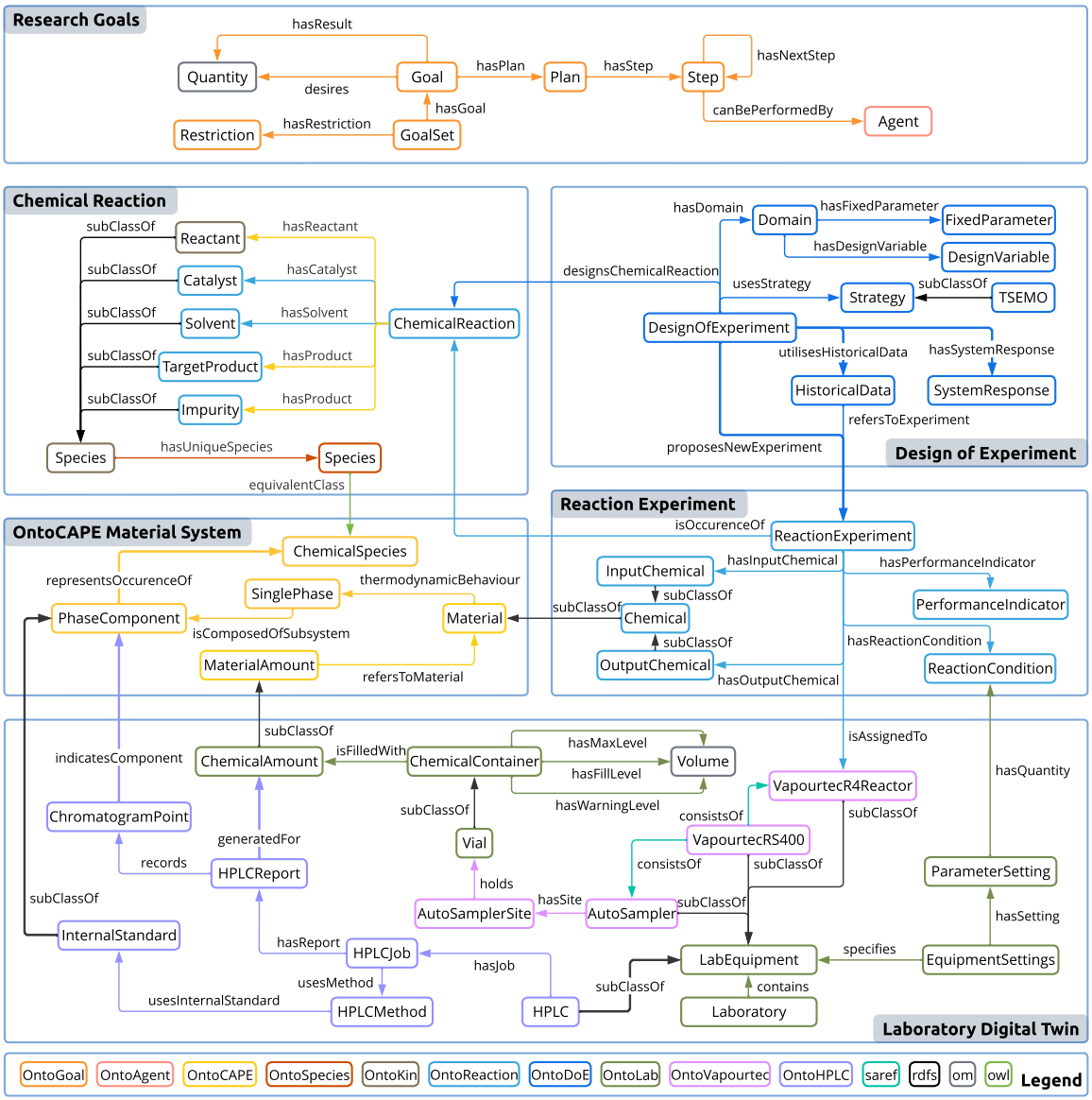


Fig. 6.2 A selection of concepts and relationships capturing different aspects in SDLs. Their namespaces correspond to the colour coding. For namespace definitions see Appendix B.1.

OntoGoal

For closed-loop optimisation in SDLs, we draw parallels between the pursuit of optimal objectives and the reasoning cycles involved in pursuing a goal [291, 345]. Figure 6.3 presents OntoGoal[1], an ontological markup for the process of pursuing research goals by autonomous agents. It is inspired by a type of rational agent architecture, namely the

[1]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontogoal/OntoGoal.owl

Belief–Desire-Intention (BDI) [291, 292] architecture rooted in cognitive psychology theory developed by Bratman [30]. In their definition, *beliefs* are the agent's knowledge about itself and its surrounding environment, *desires* are objectives that they would like to achieve, and *intentions* are possible actions that can be adopted to achieve the committed goal. Specifically, a *goal* is an instantiation of the "desire" but with more emphasis on the achievable side and a *plan* is a concrete realisation of "intentions" as a sequence of actions.

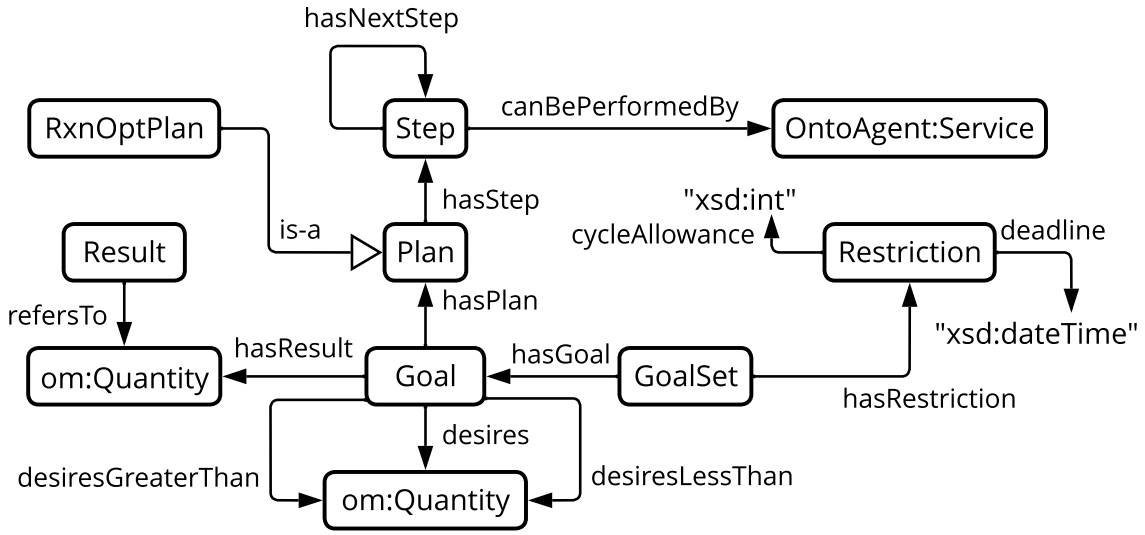


Fig. 6.3 OntoGoal ontology for research goals. The relationship with hollow arrow "is-a" represents `rdfs:subClassOf`. The remaining concepts and relationships are under the OntoGoal namespace if not stated otherwise.

Following these definitions, the multi-objective problem can be formulated as a `GoalSet` to comprise individual `Goals` (objectives). Each goal desires certain dimensional quantities and can be achieved by a `Plan` which consists of multiple `Step` that each can be performed by a corresponding agent. When the plan is executed, the state of the world changes. The beliefs, in this case, the knowledge about the reaction of interest, are updated and reflected as historical data in the next iteration. In practice, the resources available to pursue a goal are almost always limited. The current design incorporates the `cycleAllowance` and `deadline` as two limiting factors. Future development can be made to take the classic metric when bench-marking the performance of optimisation algorithms into consideration [102], *i.e.*, improvement of the hypervolume.

From the implementation perspective, this is akin to a specialised research sub-domain within the scientific workflow community that focuses on the management of iterative workflows abstracted as Directed Cyclic Graphs (DCGs) [196]. In this regard, we adopt the DIF, a KG-native approach as introduced in Chapter 4, to manage the iterative workflow.

Reuse of OntoCAPE expressions

In developing chemical ontologies for SDLs, we draw upon the lessons learnt in creating ontologies for chemical plants. One prominent example is the OntoCAPE[2] material and chemical process system [242] ontology, which describes materials from three aspects: the `ChemicalSpecies` that reflects the intrinsic characteristics, `Material` as part of the phase system which describes macroscopic thermodynamic behaviour, and `MaterialAmount` that refers to a concrete occurrence of an amount of matter in the physical world. Previously, the World Avatar has used concepts from OntoCAPE in the areas of kinetic modelling [98] and dispersion simulation [248].

Figure 6.4 illustrates relevant concepts, relationships, and instances of OntoCAPE re-used in this work to describe chemicals in a laboratory environment. Specifically, three concepts, `OntoSpecies:Species`, `OntoReaction:Chemical` and `OntoLab:ChemicalAmount`, are made as either an equivalent class or sub-class of their OntoCAPE counterparts. The rationale for not directly using the concepts from OntoCAPE is to allow other customised definitions introduced in the following sections.

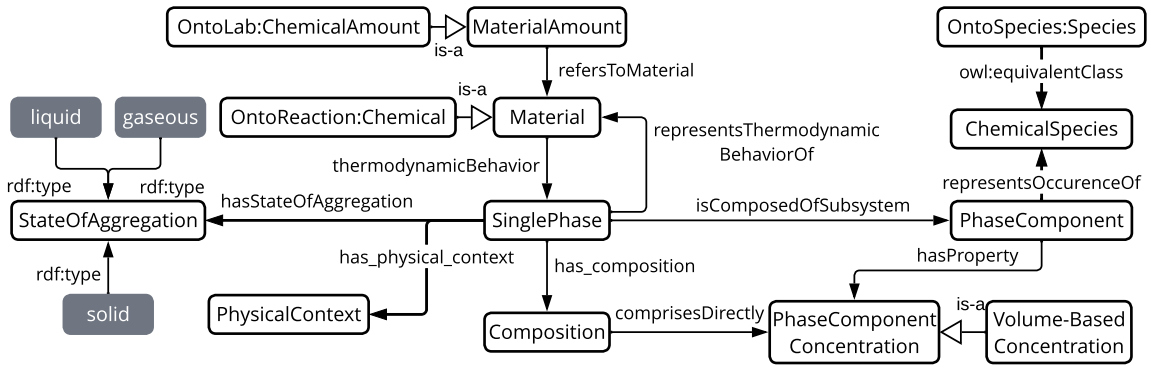


Fig. 6.4 Ontological representation from OntoCAPE re-used for chemical representation in the World Avatar. For their corresponding namespaces please see Table B.1.

OntoReaction

OntoReaction[3] is an ontology that captures conceptual descriptions of wet-lab reaction experiments. Figure 6.5 illustrates three core aspects of OntoReaction, including the reaction scheme, reaction condition and performance indicator. Notably, this ontology is to be

[2]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontocape/OntoCAPE.owl

[3]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontoreaction/OntoReaction.owl

distinguished from OntoRXN [117], which is designed for reaction networks in computational chemistry applications and is covered by OntoKin [98] and OntoCompChem [197] in the World Avatar project.

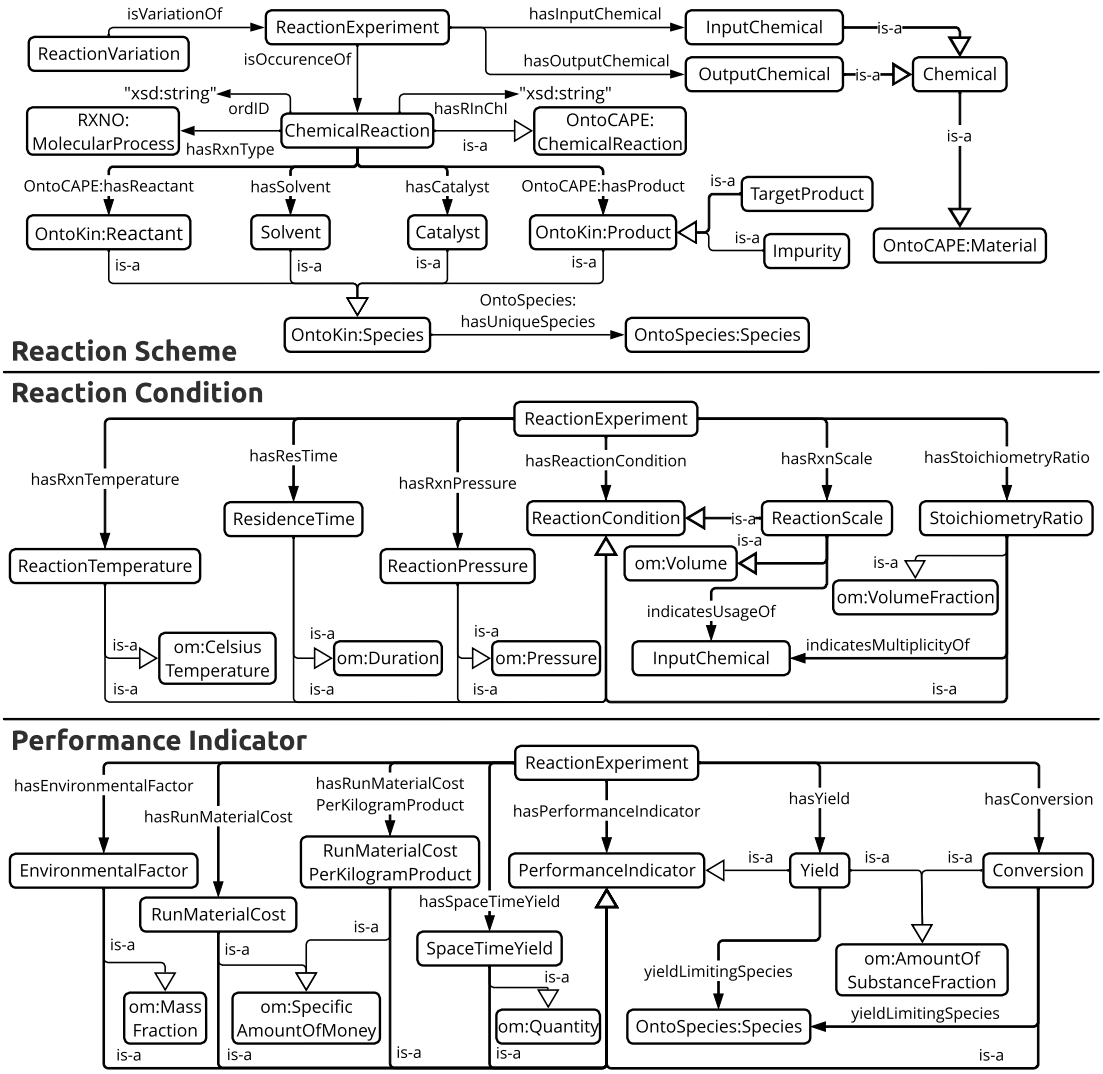


Fig. 6.5 OntoReaction ontology for chemical reaction experiment. The relationship with hollow arrow "is-a" represents `rdfs:subClassOf`. The remaining concepts and relationships are under the OntoReaction namespace if not stated otherwise. The complete IRI for concept `RXNO:MolecularProcess` reads `http://purl.obolibrary.org/obo/MOP_0000543`.

OntoReaction emphasises data utilisation and sharing between different organisations, as well as formulating the generated data into an algorithmically accessible form. Therefore, the concepts and relationships are inspired by the schema of existing chemical reaction databases, such as ORD [184] and UDM [280], and reaction data mining studies [349, 138].

The key concept in OntoReaction is the `ReactionExperiment`, which represents a concrete occurrence of `ChemicalReaction` that is sampled at a set of `ReactionConditions` and measures certain `PerformanceIndicators`. Identifiers are added to facilitate reaction search and indexing, such as `ordID` and `hasRInChI`. Other available identifiers include `hasEquation`, `hasRDFILE`, `rxnSMILES`, *etc*. Classification of reaction types can be made by linking the reaction instance to subclasses of `RXNO:MolecularProcess` [94].

To support flow chemistry applications, `Solvent` and `Catalyst` are added as subclasses of `OntoKin:Species`. This expands the World Avatar's coverage beyond gas-phase reactions. The `ReactionCondtion` and `PerformanceIndicator` are provided as generic concepts for extensions. As dimensional quantities, the extended concepts also inherit suitable subclasses of `om:Quantity` defined in the ontology of units of measure (OM) [299]. Additional reaction conditions and performance indicators relevant to other chemistry domains can be incorporated if necessary, such as those for photochemical and electrochemical reactions.

OntoDoE

OntoDoE[4] is an ontological markup designed for the conceptualisation of DoE in optimisation campaigns. In contrast to Ontology of scientific EXPeriments (EXPO) [339], which provides a comprehensive description of scientific experiments, OntoDoE provides a suitable description to capture the metadata of DoE conducted by software packages. In this regard, Ontology for numerical Design of Experiments (ODE) [25] is a relevant ontology developed for the numerical DoE, but it is not publicly available and thus direct reuse is not possible. Hence, we create our own OntoDoE.

Figure 6.6 maps the core concepts and relationships of OntoDoE, which follows the abstraction adopted by Garud et al. [120]. To construct a DoE study, a domain needs to be defined. It can comprise both design variables, either continuous or categorical, and fixed parameters. The primary objective of a DoE study is to suggest a new set of conditions in the search space that optimises the desired system responses based on available historical data. This process may involve different sampling strategies, which are highly relevant to the chosen modelling tools. In this work, we employ the Python package `summit` [102], and thus OntoDoE also reflects its available algorithms and the naming conventions of its data classes.

OntoDoE is intended to be used in conjunction with OntoReaction. Specifically, instances of `OntoReaction:ReactionExperiment` are used to represent both historical or new experiments. If no prior data is available, OntoDoE can still be used by providing a DoE instance as a template for the `OntoReaction:ChemicalReaction` to be optimised.

[4]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontodoe/OntoDoE.owl

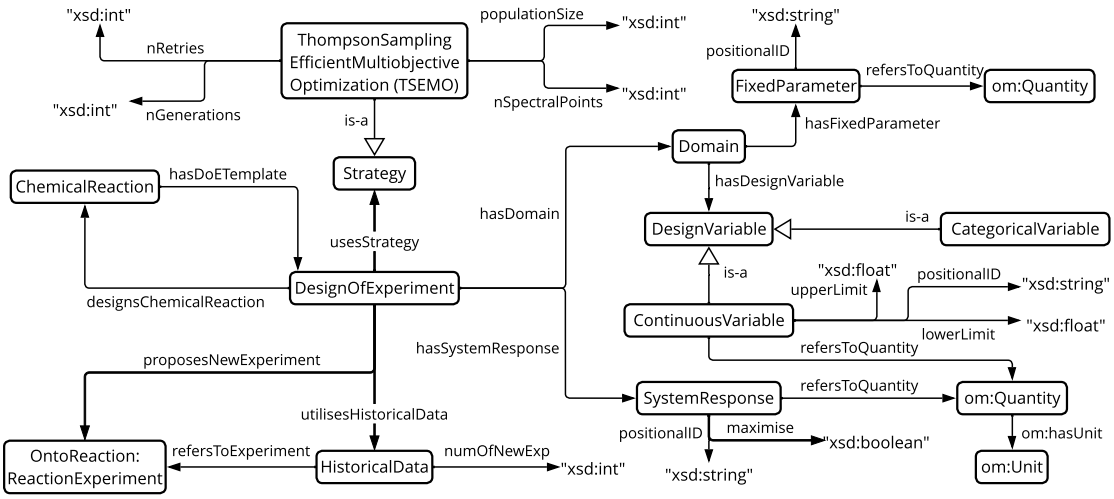


Fig. 6.6 OntoDoE ontology for the design of experiments. The relationship with hollow arrow "is-a" represents `rdfs:subClassOf`. The remaining concepts and relationships are under the OntoDoE namespace if not stated otherwise.

OntoLab

In the development of our hardware ontologies, we have expanded upon concepts from the Smart Applications REFerence (SAREF) ontology [66], which is widely adopted in the field of the Internet of Things (IoT). We introduce OntoLab[5] to represent the digital twin of a laboratory. The current iteration focuses on the functional aspects of the laboratories, including laboratory equipment and chemical containers as illustrated in Fig. 6.7. The geospatial and visualisation aspects of laboratories are omitted for simplification. To the best of our knowledge, there is no single ontology for laboratory environments that is readily available and fits our purpose. Therefore, concepts from different ontologies are utilised wherever suitable.

We describe a piece of `LabEquipment` from three perspectives, including its static specifications, the dynamic configuration for actuation, and the measurements that can be acquired from its sensors. The inspiration comes from the process of abstracting relevant information on the hardware involved in Jeraal et al. [178]. The specifications are normally dimensional quantities, *e.g.*, height, width, and price, which we use concepts from OM. The other two perspectives align with the SAREF ontology. We thus define the `LabEquipment` as a subclass of `saref:Device`, which allows inheriting other useful concepts and relationships provided in the SAREF ontology. The dynamic configuration part connects the abstract-level data expressed in OntoReaction with concrete equipment realisation by translating the

[5]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontolab/OntoLab.owl

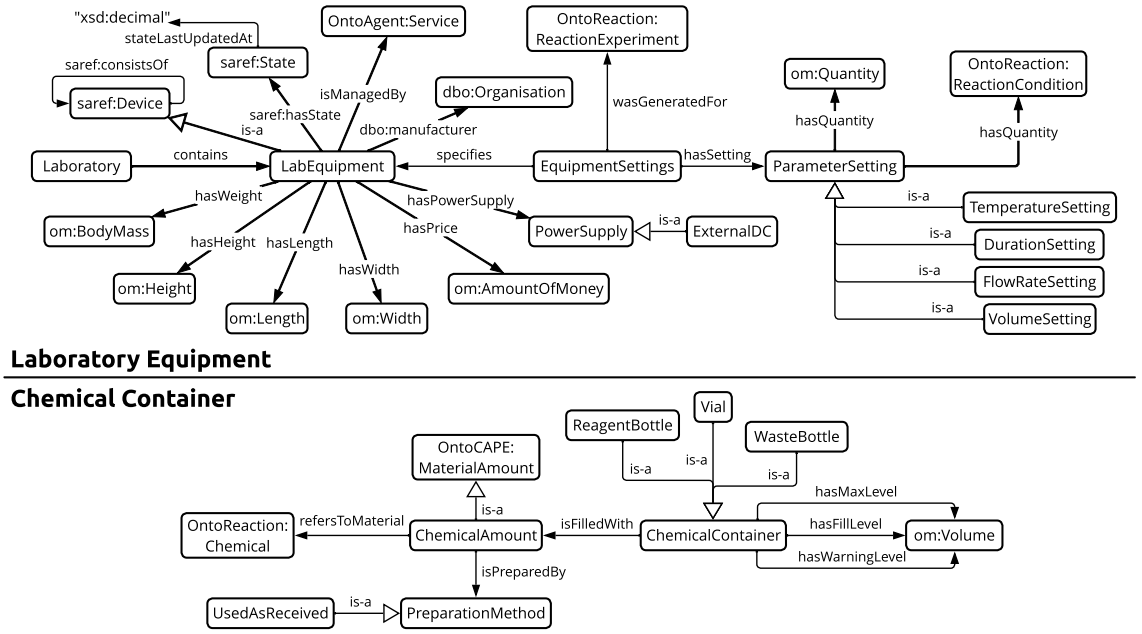


Fig. 6.7 OntoLab ontology for laboratory digital twin. The relationship with hollow arrow "is-a" represents `rdfs:subClassOf`. The remaining concepts and relationships are under the OntoLab namespace if not stated otherwise.

reaction conditions to `ParameterSetting` and further assembling `EquipmentSettings` for corresponding lab equipment. The measurement obtained from the lab equipment varies across kits that serve different purposes. In practice, both configuration and data collection are done by the software agent that manages each piece of equipment. For those designs please refer to later sections.

A few relevant concepts exist in OntoCAPE for describing containers, however, they are defined at the level of a chemical plant and also not generic enough to cover the different types of containers in a lab environment. We thus create `ChemicalContainer` class. Following a similar naming convention, `ChemicalAmount` is proposed to represent the physical existence of chemicals in the container. Different quantitative amounts can be attached to track the consumption and refill of the chemicals.

OntoVapourtec

OntoVapourtec[6] is an ontological developed for the Vapourtec flow chemistry system used in this work. The core concepts (see Fig. 6.8) are the individual equipment that extends `OntoLab:LabEquipment`, *i.e.*, the integrated system `VapourtecRS400` that consists of tube

[6]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontovapourtec/OntoVapourtec.owl

reactor, pumps, and optionally an autosampler depending on how the chemicals are sourced. The configuration of the equipment relies on available APIs. Each parameter will be translated to construct the final experiment file for execution. More details are covered in section 6.2.3.

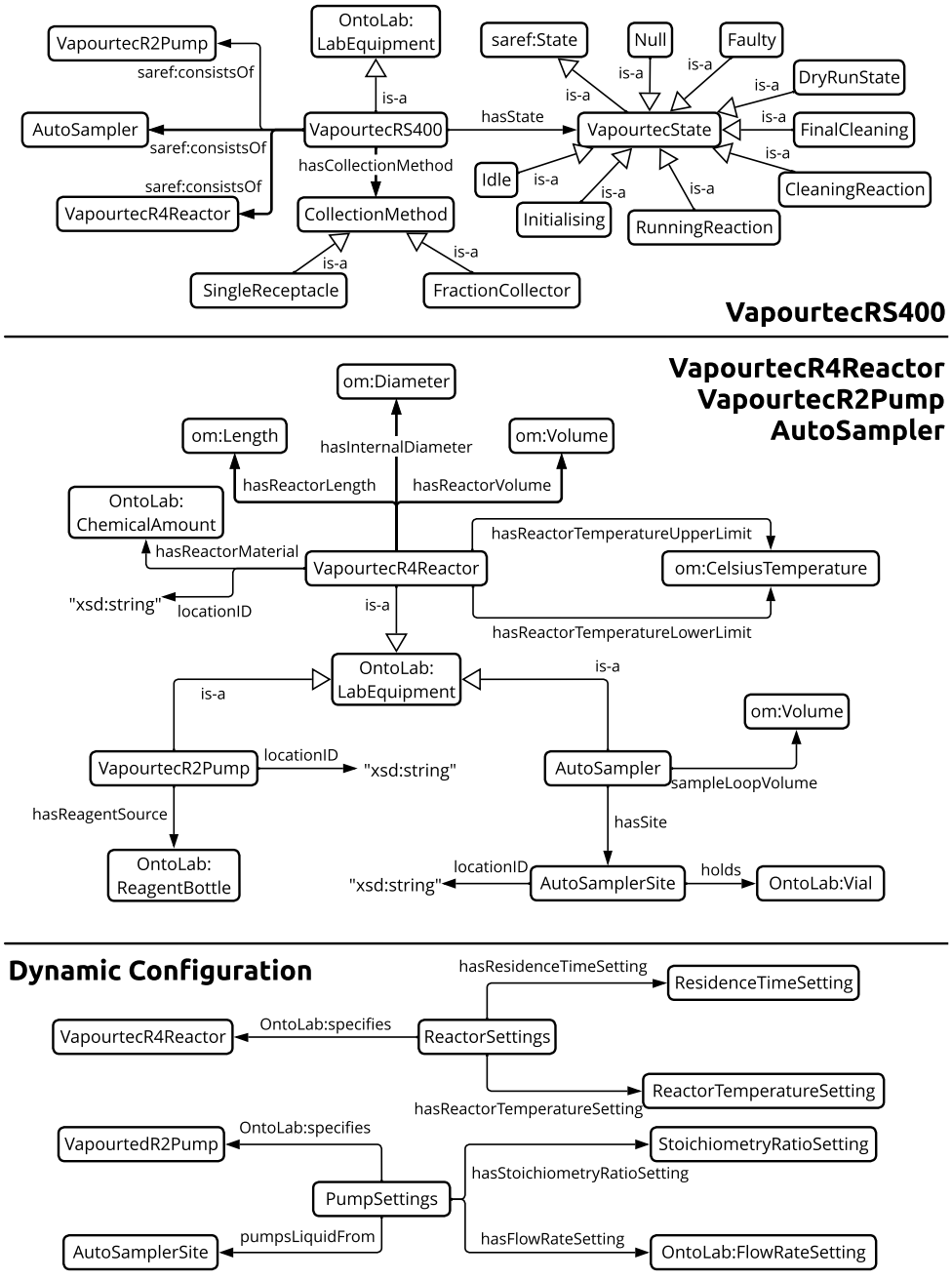


Fig. 6.8 Core concepts and relationships in OntoVapourtec ontology for Vapourtec flow chemistry platform. The relationship with hollow arrow "is-a" represents `rdfs:subClassOf`. The remaining concepts and relationships are under the OntoVapourtec namespace if not stated otherwise.

OntoHPLC

OntoHPLC[7] is designed to provide a simplified representation of High-Performance Liquid Chromatography (HPLC), as illustrated in Fig. 6.9. Its goal is not to provide a full-fledged representation of HPLC, as the Vapourtec `FlowCommander` already "automates" the HPLC analysis by sending a serial command at the peak of the reaction steady-state stream as a trigger for sample injection. Instead, OntoHPLC demonstrates the proof-of-concept for the dKG approach with a focus on `ChromatogramPoints` captured in the `HPLCReport`. The components presented in the reaction outlet stream can be identified by matching the retention time at which the peak appears with those documented in the `HPLCMethod`. Moreover, the raw HPLC reports are also preserved in the dKG via `remoteFilePath`, which points to a remote file server for easy human inspection.

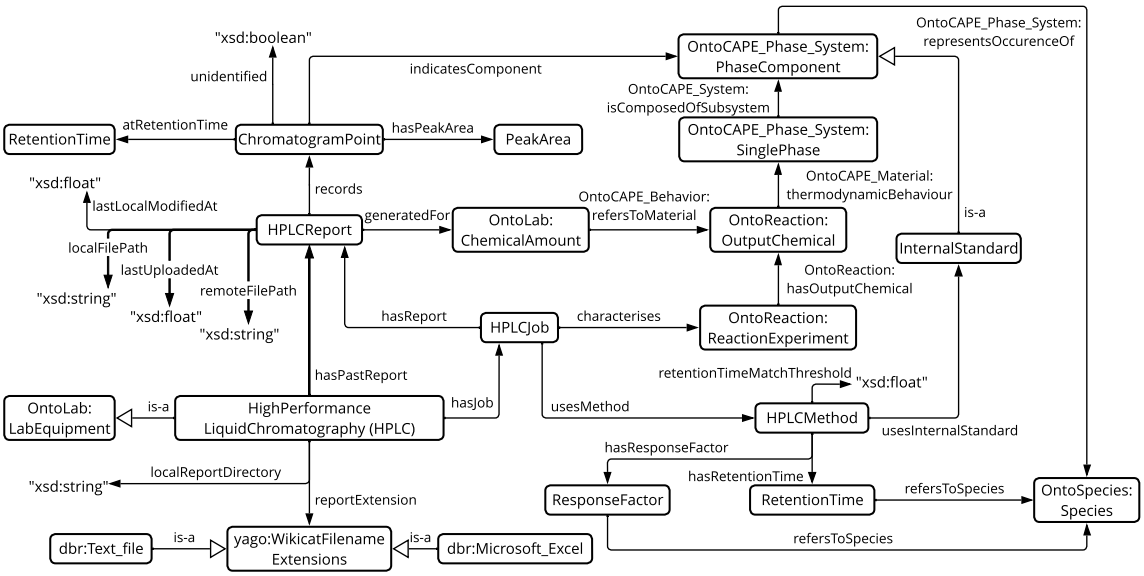


Fig. 6.9 OntoHPLC ontology for HPLC. The relationship with hollow arrow "is-a" represents `rdfs:subClassOf`. The remaining concepts and relationships are under the OntoHPLC namespace if not stated otherwise.

Contextualised reaction informatics

By utilising ontologies as blueprints, we can instantiate reaction information while preserving connections to contextual recordings. Taking the distributed SDLs use case as an example, it involves an aldol condensation reaction between benzaldehyde **1** (bold number for

[7]https://raw.githubusercontent.com/cambridge-cares/TheWorldAvatar/main/JPS_Ontology/ontology/ontohplc/OntoHPLC.owl

reference) and acetone **2**, catalysed by sodium hydroxide **3** to yield the target product benzylideneacetone **4** [178], which is pharmaceutically relevant that can be used to treat idiopathic vomiting as an NK-1 receptor inhibitor [272]. Additionally, reported side products include dibenzylideneacetone **5** and further condensation products from acetone polymerisation.

Figure 6.10 illustrates the chosen reaction in the dKG as viewed through various roles within a laboratory, each with its unique perspective. Taking the starting material benzaldehyde as an example, it demonstrates how a dKG can enhance the daily work of different roles. A chemist, more interested in conceptual description, might look at benzaldehyde as a reactant and search for relevant species information. A data scientist might examine its concentration to determine the appropriate usage of other chemicals when designing conditions for a particular reaction experiment. Meanwhile, the 3D digital twin built on top of the dKG offers a lab manager a centralised hub for real-time monitoring of lab status [286], ensuring the availability of an internal standard that can be mixed with the benzaldehyde during the actual execution of the experiment for characterisation. In practice, the same individual might play several roles, and the emphasis here is on the cross-domain interoperability facilitated by the amalgamation of different aspects into a unified dKG. This integration ensures the relevance of information to a diverse range of users while maintaining human oversight. Consequently, this approach may present opportunities for the enhancement of various digital applications, such as the utilisation of virtual reality for laboratory training [86].

The integration of chemical knowledge from PubChem, represented by OntoSpecies for unique species identification [273], serves as a critical link between these facets of chemicals. It enables the identification of potential input chemicals based on the reactant and solvent during DoE and allows for the selection of appropriate sources of starting materials from multiple chemical containers. By combining various aspects into a unified representation, dKG encompasses information that is pertinent to diverse users while keeping humans in the loop.

This approach enables flexibility and extensibility in the system. As the digital twin of each lab is represented as a node in the dKG, new hardware can be added or removed during the optimisation campaign by simply modifying the list of participating laboratories. The experimental allowance can also be updated when more chemicals become available. The system also supports data sharing across organisations at the very moment the data are generated.

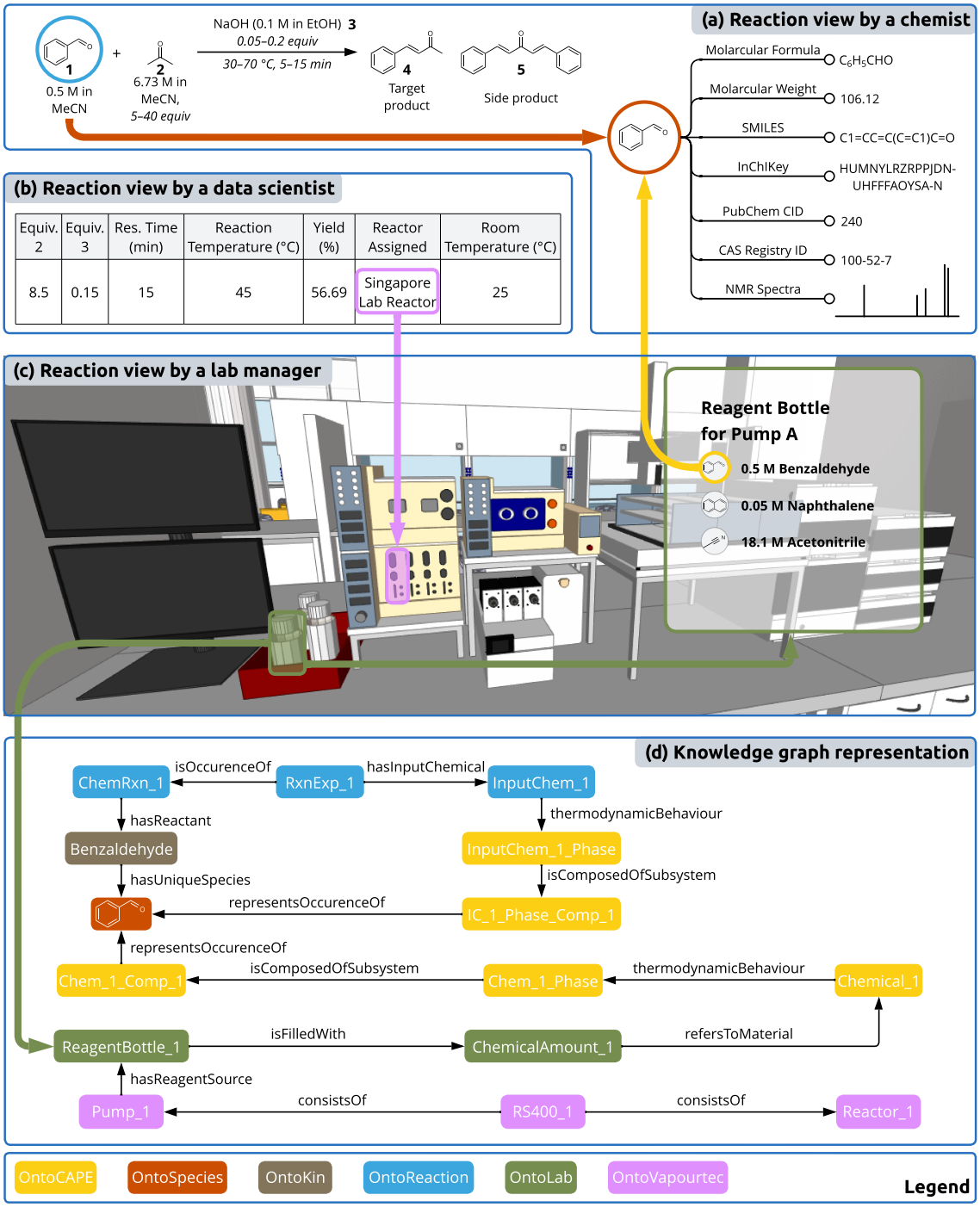


Fig. 6.10 A snapshot of reaction views from different perspectives. (a) A chemist view of a reaction is based on the chemical structures. (b) A data scientist view of a reaction is based on the experiment conditions and resulting performance indicators. (c) A lab manager view of a reaction is based on hardware status and chemical availability. (d) The KG representation puts chemical informatics into context, allowing for queries and answers from different layers of abstraction (views). The colour coding corresponds to the ontological expression.

6.2.3 Goal-driven knowledge dynamics

This section describes the dynamicity of the dKG which is enabled by the presence of software agents. Firstly, we provide a brief overview of the DMTA cycles formulated as information flow facilitated by goal-driven agents. We then delve into the internal logic of each of the agents involved. We also present their distributed deployment that is networked via the dKG. The section concludes by showcasing the derived information stepping process that snapshots different stages in the workflow.

Information flow of DMTA cycles

Figure 6.11 presents a high-level overview of the goal-driven self-evolution of the dKG during closed-loop optimisation. The process begins with goal derivation when the scientist initiates a goal request. The Reaction Optimisation Goal (ROG) Agent translates the goal request into a machine-readable statement that captures the scientist's intention. Each objective is considered a reward function for the agents' operations and grouped into a goal set. For each participating laboratory, a `Goal Iteration Derivation` instance is instantiated in the dKG and is requested for execution by the Reaction Optimisation Goal Iteration (ROGI) Agent.

The goal iteration stage plays a central role in self-evolution, where the ROGI Agent initiates the flow of information among the participating agents towards achieving the goal set. This process begins with ROGI Agent creating tasks for the corresponding agents involved in the DMTA cycle, including the DoE Agent, Schedule Agent, and Post-Processing Agent. The DoE Agent perceives the dKG to retrieve prior data and chemical stock available for experiments and then proposes a new experiment. Based on the proposed conditions, the Schedule Agent selects the most appropriate hardware to execute the experiment from those available in the participating laboratory. This is accomplished by generating tasks for the agents managing the selected digital twin. These agents actuate the equipment in the physical world to perform reaction and characterisation. When the HPLC report is generated, the Post-Processing Agent analyses the chromatogram data to calculate the objectives.

During the third stage, the ROG Agent utilises the obtained results to determine whether the next iteration should be pursued. To do so, it checks if the optimal value(s) fulfils the pre-defined goal(s) and if the resources are still available. The reaction experiment performed in the current iteration is then considered historical data to be included as input for the succeeding round of the `Goal Iteration Derivation` across all participating SDLs. Afterwards, a new request will be made to the ROGI Agent to start a new iteration, forming a self-evolving feedback loop.

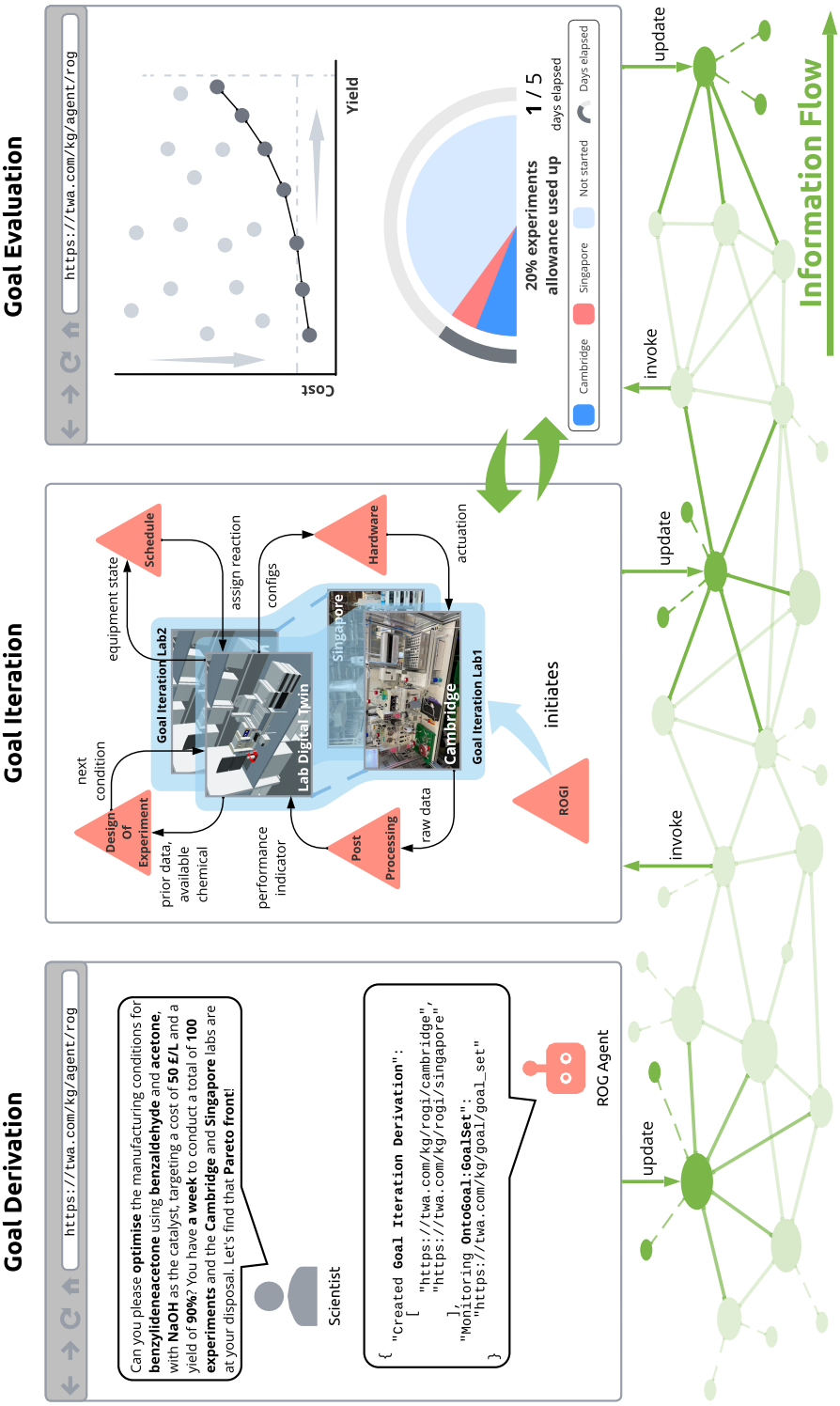


Fig. 6.11 Autonomous workflow triggered in response to goal requests from scientists as information travels within the dKG.

Internal logic of agents

To ensure correct data dependencies and the order of task execution, we employed DIF to manage the iterative workflow. We implemented each software agent using the `DerivationAgent` template provided by the Python version of the framework. Specifically, agents utilise the asynchronous communication mode when interacting with the dKG as conducting experiments is inherently a time-consuming process. Each of the agents monitors the jobs assigned to itself and records the progress of execution in the dKG. As agents modify the dKG and subsequently actuate the real world autonomously once active, it is important to make sure they behave as expected. In this regard, unit and integration tests are provided to help with responsible development.

DoE Agent The DoE Agent (v1.2.0[8]) configures the metadata of DoE studies using `OntoDoE:DesignOfExperiment` and to locates the SDL where the experiment is expected to be carried out using `OntoLab:Laboratory`. The relevant information is queried from the dKG and used to form Python objects required by the `summit` package [102]. In this iteration, the TSEMO [27] algorithm is used to propose the next experiment condition. The suggestions are then parsed to construct a Python object of `OntoReaction:ReactionExperiment` and translated to triples. Notably, the possible `OntoReaction:InputChemical` is located based on the chemicals present in the chemical containers available in the specified laboratory. DIF populates these statements back into the dKG and connects the IRI of the created experiment as input to the next steps in the workflow. A detailed UML activity diagram is provided as Fig. B.1.

VapourtecSchedule Agent The Vapourtec Schedule Agent (v1.2.0[9]) schedules execution of experiments in multiple laboratories located in different geographical areas. The experiment to be carried out and the target laboratory for experimentation are defined by the input instances of `OntoReaction:ReactionExperiment` and `OntoLab:Laboratory`, respectively. The scheduler uses the First In First Out (FIFO) algorithm, also known as First Come First Serve (FCFS), for assigning the reaction experiment to the digital twin of a suitable reactor that has the required chemicals and is capable of conducting the reaction conditions. This is done by creating the job requests (derivations) for the agents responsible for managing the hardware. The scheduler agent monitors the progress of the experiment by periodically checking if the new HPLC report is generated. The new report is then passed

[8]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/DoEAgent

[9]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/VapourtecScheduleAgent

on to the post-processing step as input. More sophisticated scheduling algorithms may be implemented in future versions. A detailed UML activity diagram is provided as Fig. B.2.

Vapourtec Agent The Vapourtec Agent (v1.2.0[10]) liaises between the physical setup and its digital counterpart by surrounding the software provided by the hardware vendor, *i.e.*, `FlowCommander` (v1.12). The primary usage of `FlowCommander` is for manual control and inspection, therefore, only limited high-level functions are provided by its API. The only data that can be programmatically retrieved from the hardware is the state of the entire system. These states indicate the stage of the reaction execution, *e.g.*, initialising, running reaction, cleaning, *etc.*, and are updated in the dKG every 30 seconds. The live data collected from the temperature and pressure sensors are only displayed in the graphical interface of `FlowCommander`.

Upon an assigned `OntoReaction:ReactionExperiment`, the Vapourtec Agent translates the reaction conditions to a list of equipment settings to configure the Vapourtec equipment it manages. This enables tailored pump settings when locating the inlet streams that match with the concentrations of `OntoReaction:InputChemical` as we use different sourcing methods in the two labs, *i.e.*, autosampler in Cambridge and reagent bottles in Singapore. Once all configurations are prepared, the agent compiles a CSV file and saves a copy in the file server for the record. The command for execution is sent when the hardware is idle to ensure no interruption to the cleaning steps of the previous experiment.

During the reaction, the agent updates the liquid level of the chemical containers in the digital twin. Together with the labelled warning level, this allows the agent to only select the vials that still have enough chemicals in the following reactions, *i.e.*, those above the warning level, as well as inform researchers if any containers need a refill.

As the reactor is physically tubed to the HPLC via a four-way Valco Instruments Company Incorporated (VICI) switching valve, `FlowCommander` is configured with an external analysis trigger to time the sample injection. The material flow for the reaction and characterisation will run in the physical world without any further intervention required. The agent then creates an instance of `OntoLab:ChemicalAmount` referring to the reactor outlet. Populating its triples to the dKG and assigning its IRI as the input to the HPLC Agent are again managed by DIF. A detailed UML activity diagram is provided as Fig. B.3.

[10]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/VapourtecAgent

HPLC Agent The HPLC Agent (v1.2.0[11]) monitors HPLC reports and uploads newly generated reports to the dKG. The purpose of this agent is not to provide complete control of HPLC analysis, as it is physically triggered by the injection from the four-way VICI switching valve. We refer interested readers to Nambiar et al. [255] for codes on a more comprehensive control of HPLC. The connection between the newly generated report and the reaction experiment it characterises is initiated by the job request once the instance of `OntoLab:ChemicalAmount` is instantiated by the Vapourtec Agent. If a new HPLC report is generated after the analysis has commenced, a new instance of `OntoHPLC:HPLCJob` is constructed and populated back to link the relevant instances. A detailed UML activity diagram is provided as Fig. B.4.

HPLCPostPro Agent The HPLCPostPro Agent (v1.2.0[12]) post processes the instances of `OntoHPLC:HPLCReport` by utilising a hypothetical model of the reactor and its inlet/outlet streams. Firstly, the agent constructs the reactor model and its inlet streams. It then identifies the chemical species from the raw HPLC report and calculates their concentrations to form the reactor outlet. The performance indicators currently available include yield, conversion, run material cost, Space-Time Yield (STY), and Environmental factor (E-factor).

Figure 6.12 provides the abstraction adopted in the hypothetical models used for post-processing. The data stored in the ontological objects is transferred to these models and employed to compute the performance indicators. The class of `DimensionalQuantity` is employed to support automated unit conversion when writing the calculated objectives back to the dKG.

The logic of the species identification from the raw HPLC report follows below steps:

1. Iterate through the list of peaks and remove any peaks that are outside the threshold range of available retention time for all species, labelling these peaks as unidentified.

2. For the remaining peaks that fall within the retention time threshold range, identify the peak with the largest area for each species and mark any unselected peaks as unidentified.

3. Calculate the concentration of each identified species by using the response factor and the corresponding peak area.

[11]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/HPLCAgent

[12]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/HPLCPostProAgent

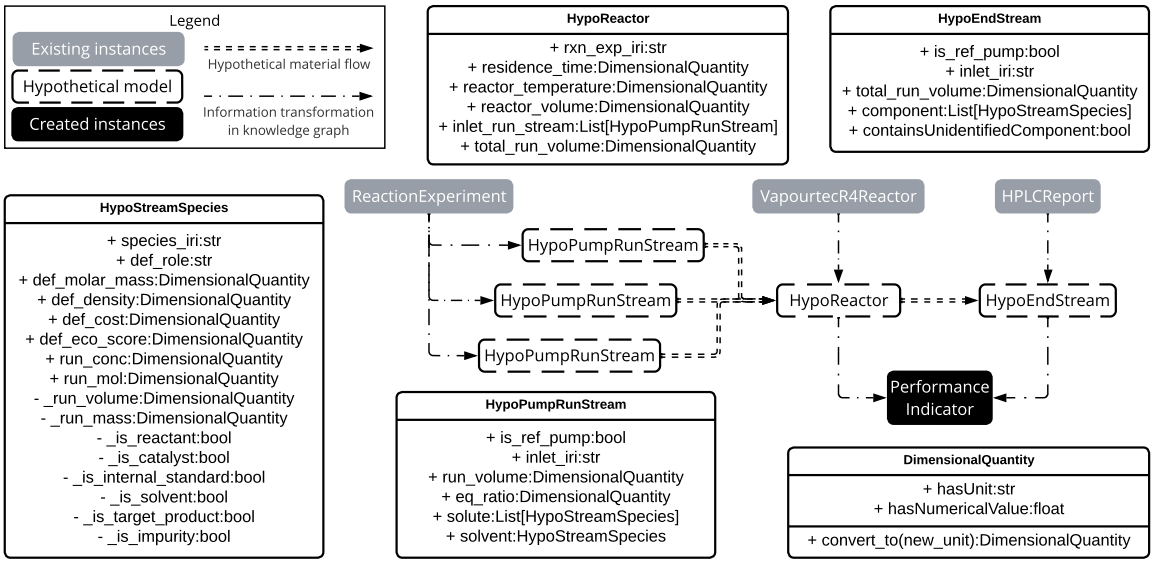


Fig. 6.12 Hypothetical models for post-processing HPLC reports.

With the concentrations identified in the reaction end stream, the objectives can be calculated following the below equations:

$$\text{Yield} = \frac{\text{Actual } n_{\text{product}}}{\text{Theoretical } n_{\text{product}}} \times 100\% = \frac{\text{Actual } c_{\text{product}}}{\text{Theoretical } c_{\text{product}}} \times 100\%, \tag{6.1}$$

where n_{product} refers to the number of moles of product produced and c_{product} refers to its concentration in the reaction stream. It should be noted that the theoretical amount that could be produced is calculated based on the limiting reactant.

$$\text{Conversion} = \frac{\text{Initial } n_{\text{limiting reactant}} - \text{Final } n_{\text{limiting reactant}}}{\text{Initial } n_{\text{limiting reactant}}} \times 100\%, \tag{6.2}$$

where $n_{\text{limiting reactant}}$ is the amount of limiting reactant presented in the reaction mixture.

$$\text{Run Material Cost} = \frac{\sum_{i=1}^{k_{chemicals}} C_i V_i}{V_{\text{reaction scale}}}, \tag{6.3}$$

where C_i is the cost of chemical i sourced from the pumps, V_i is the volume of chemical i used in the reaction, $k_{\text{chemicals}}$ is the number of chemicals sourced from the pumps (excluding the internal standard), and $V_{\text{reaction scale}}$ is the reaction scale for the pump that contains primary starting material, *i.e.*, benzaldehyde.

$$\text{STY} = \frac{m_{\text{product}}}{V_{\text{reactor}} \times t_{\text{res}}}, \tag{6.4}$$

where m_{product} is the mass of the product, V_{reactor} is the reactor volume, and t_{res} is the residence time.

$$\text{E-factor} = \frac{m_{\text{waste}}}{m_{\text{product}}}, \tag{6.5}$$

where m_{waste} is the mass of waste generated and m_{product} is the mass of product produced. Note that if no product is produced, the E-factor will be assigned an infinity value.

ROGI Agent The ROGI Agent (v1.2.0[13]) oversees each DMTA cycle involved in the iterative pursuit of research goals. In each iteration of the closed-loop optimisation process, the ROGI Agent initiates by creating an instance of `OntoDoE:DesignOfExperiment` by converting the specified goals in the `OntoGoal:GoalSet` into design objectives and filtering the prior results of `OntoReaction:ReactionExperiment` to compile the object of historical data. It then generates a sequence of job requests that consists of designing, scheduling, and post-processing a new experiment. As the workflow progresses, the dKG automatically evolves. The ROGI Agent periodically monitors whether the objectives are computed and marks them as `OntoGoal:Result`, which is used to determine whether to proceed to the next round of optimisation. It is worth noting that an instance of `Goal Iteration Derivation` is created per instance of `OntoLab:Laboratory` involved in the optimisation campaign. This design enables each SDL to function independently, but they can collaborate in case multiple SDLs are participating. A detailed UML activity diagram is provided as Fig. B.6.

ROG Agent BDI agent architecture and consequently multi-agent system is well studied in the literature and its development has been surveyed in [26, 72]. There are different types of goals: test goals, achievement goals, and maintenance goals. In this study, the reaction optimisation problem can be seen as setting targets for achievement goals. Each step made by the agent towards achieving the goal can be seen as agents making rational decisions based on the status of its environment, *i.e.*, the dKG. As the system interacts with the physical world, it is also important to set resources and deadlines for goals. In the current development,

[13]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/RxnOptGoalIterAgent

these settings are provided by human researchers as a goal request from a web front end and subsequently handled by the ROG Agent (v1.0.0[14]).

The two most important functionalities of the ROG Agent are handling the goal request (Fig. B.6(a)) and monitoring the subsequent iteration progress (Fig. B.6(b)). Upon receiving the goal request from the researchers, the ROG Agent first validates all request parameters and then translates them to actionable ontological representations based on concepts defined in OntoGoal, *e.g.*, instance of `OntoGoal:GoalSet`. It then queries the dKG to identify historical reaction experiments for the chemical reaction specified in the goal request. Should the current best-performing reaction experiment fail to meet the goal, the ROG Agent instantiates a `Goal Iteration Derivation` per `OntoLab:Laboratory` which will later be picked up by the ROGI Agent to orchestrate the actual workflow of the reaction experiment. The ROG Agent also adds a periodical job to monitor the currently active goal set to determine if one iteration is finished. If the goals have not been achieved and the resources are still available, the ROG Agent shares the latest results with all ongoing `Goal Iteration Derivation` instances. A request will then be sent to those finished derivations to trigger the next round of reaction. This ensures minimum downtime of the equipment and enables data sharing among all the involved laboratories.

Throughout the iterative process, the ROG Agent notifies developers of the latest progress. An example email when it enters the next iteration of the goal pursuit is provided as Fig. B.7.

Distributed deployment over the internet

Taking inspiration from remote control practices in lab automation [106, 290, 44] and deployment practices commonly used by cloud-native applications, the dKG is deployed span across the internet and LAN. In the distributed SDLs use case, as illustrated in Fig. 6.13, the triplestore and file server containing the knowledge statements are hosted on a cloud platform with password-protected endpoints. The agents managing the hardware, *i.e.*, Vapourtec and HPLC Agent are locally deployed in each lab for security reasons. The remaining agents are deployed across labs and a third location to demonstrate the decentralised design. All agents need an internet connection to access the dKG to register their OntoAgent instances at start-up and then act autonomously should tasks be assigned to them. Altogether, these agents form a distributed network that facilitates the transfer of information within cyberspace, as well as actively reflects and influences the state of the physical world.

Unlike many other remote control implementations, the design of DIF eliminates direct agent-to-agent communication, which would otherwise require exposing an internet-

[14]https://github.com/cambridge-cares/TheWorldAvatar/tree/dc4bf7fb23a9209b460c94013080a979f7bb3174/Agents/RxnOptGoalAgent

accessible server on the lab's desktop – often considered dangerous by IT staff. By contrast, all docker containers are deployed in the only lab involved for the Suzuki coupling use case to improve response times and save bandwidth. The docker images of agents are available in the public registry of the World Avatar on GitHub[15] and their deployment instructions are available in this folder[16].

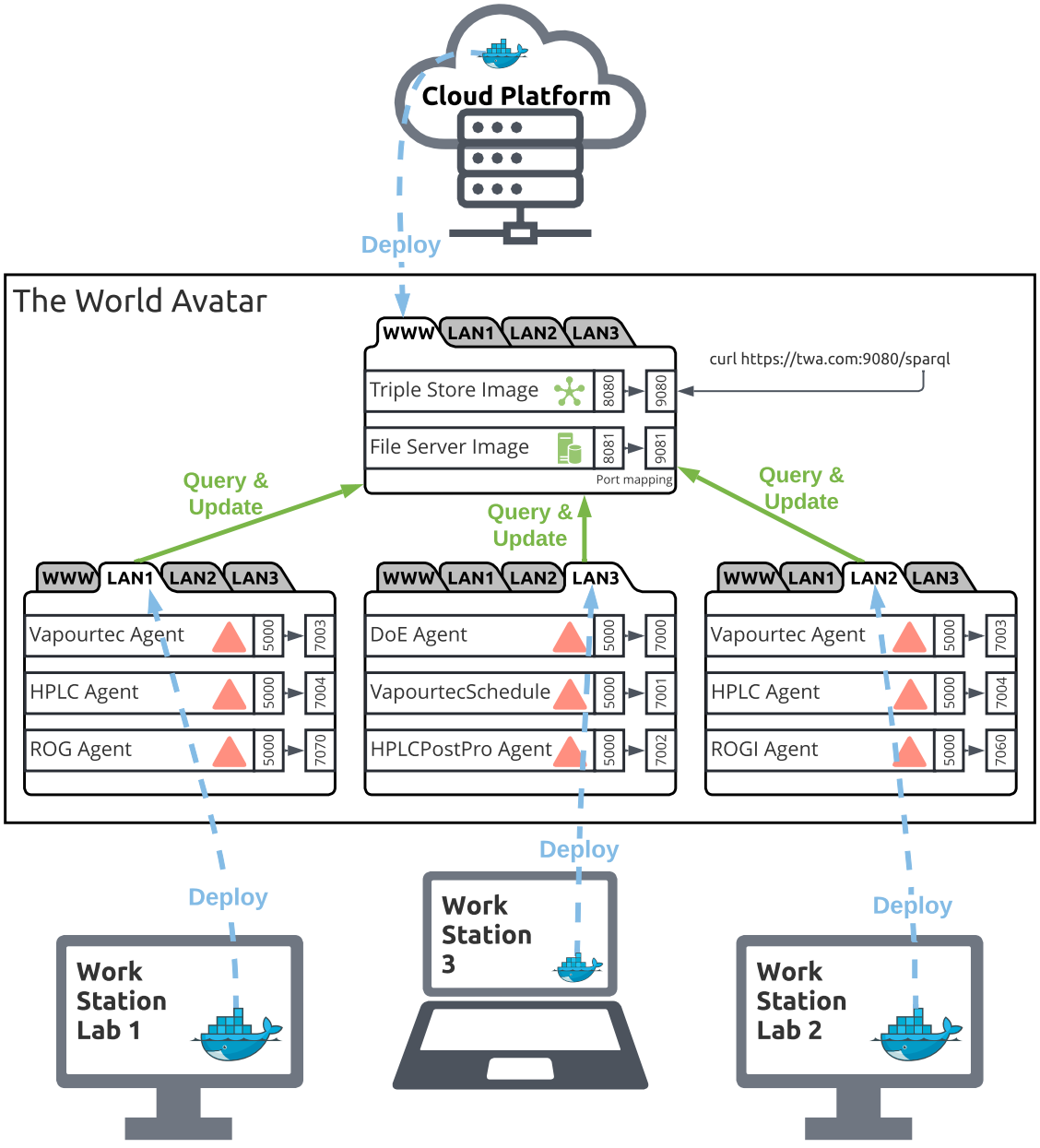


Fig. 6.13 Schematic of the distributed deployment philosophy adopted in the World Avatar.

[15] https://github.com/orgs/cambridge-cares/packages
[16] https://github.com/cambridge-cares/TheWorldAvatar/tree/main/Deploy/pips

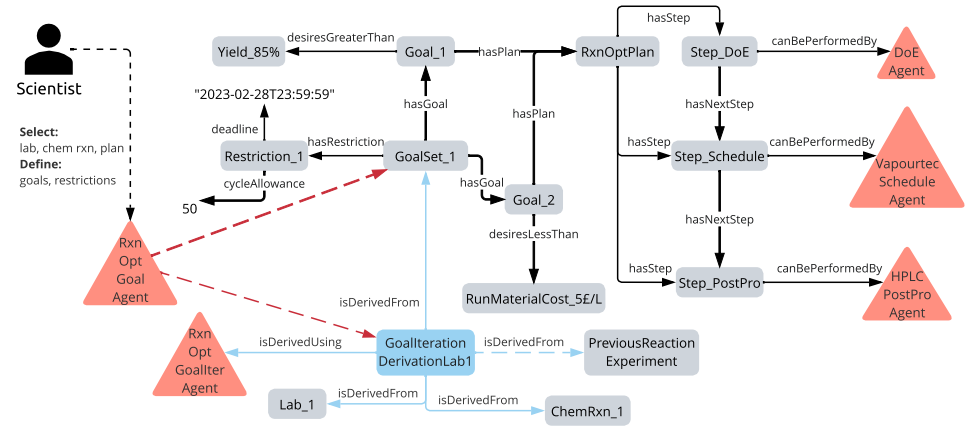
Derived information stepping

Figure 6.14 illustrates the technical implementation of dKG during the three stages of goal-driven agent dynamics. Figure 6.14(a) illustrates the initialisation of the optimisation campaign upon the goal request by the scientist. The ROG Agent parses the resources and restrictions set for the optimisation and formulates a goal set to include all objectives. Each goal is pointed to an optimisation plan which consists of a sequence of agents' actions, *i.e.*, steps in an experimental workflow. A `Goal Iteration Derivation` instance is created per laboratory and requested for execution by ROGI Agent. Any available reaction results will be used as historical data.

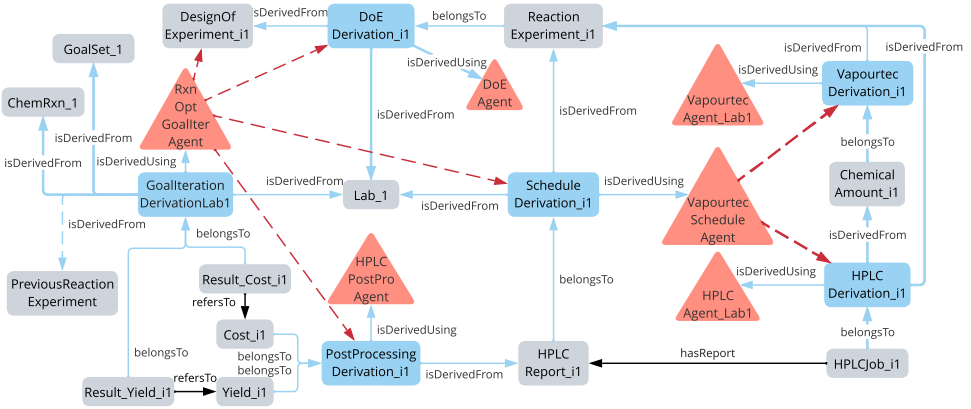
Figure 6.14(b) presents the information flow between the agents within one iteration of pursuing the goal set. It starts with ROGI Agent parsing its inputs to formulate a `DesignOfExperiment` instance and creating tasks for corresponding agents listed in the optimisation plan, involving `DoE Derivation`, `Schedule Derivation`, and `PostProcessing Derivation`. The DoE Agent takes in the metadata for the DoE and proposes a new experiment. It is then scheduled by the Vapourtec Schedule Agent to the relevant hardware contained in the specified laboratory via creating tasks for the set of agents managing the hardware, *i.e.*, `Vapourtec Derivation` for Vapourtec Agent and `HPLC Derivation` for HPLC Agent. These conditions are then populated to configure and actuate the hardware in the physical world. Once the HPLC report is generated, the HPLCPostPro Agent post-processes the chromatogram peaks to compute the objectives. These values are then attached to the output of `Goal Iteration Derivation` as the latest results.

Figure 6.14(c) sketches the process of goal evaluation when determining the next steps after each iteration. The latest results are compared with the research goals by the ROG Agent to decide if the iteration should be progressed into the next round. If the goals are not met and resources are still available, the executed reaction experiment in this iteration will be attached as input to the `Goal Iteration Derivation`. ROG Agent will then issue a new request to the ROGI Agent for another iteration and update the allowance accordingly. This process repeats until either all goals are met or all resources are consumed.

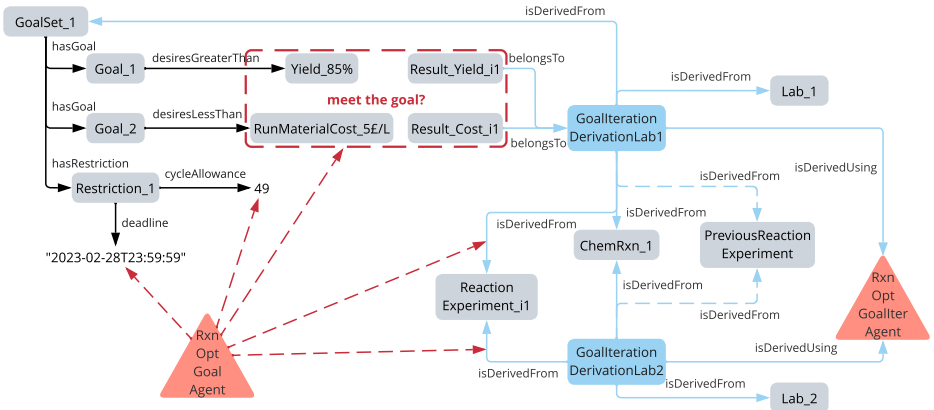
The evolution of the derivation subgraph when performing a DMTA cycle in one laboratory is exemplified in Fig. 6.15.



(a) Initialisation of goal iteration based on goal request from scientists.

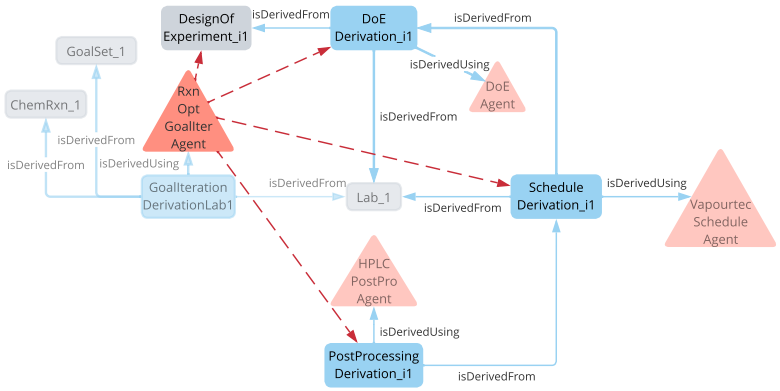


(b) Experiment workflow in one iteration of progressing towards the goal.

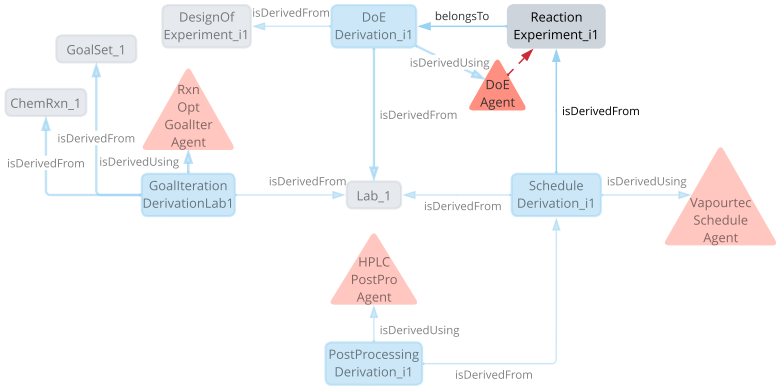


(c) Evaluation procedure when determining whether to progress to the next iteration.

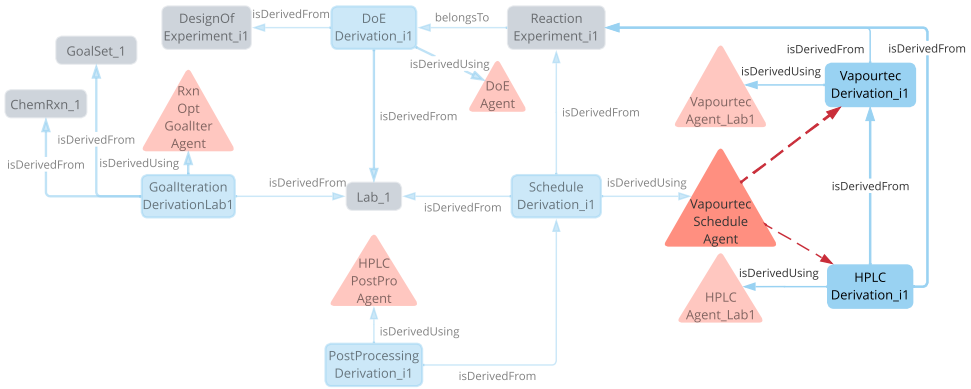
Fig. 6.14 Instantiation, iteration, and evaluation of the goal request from the scientist. Tasks in the workflow are denoted as "derivation". The red dashed lines refer to instantiation by the respective agent. The links to previous experiment results only exist should historical data be available.



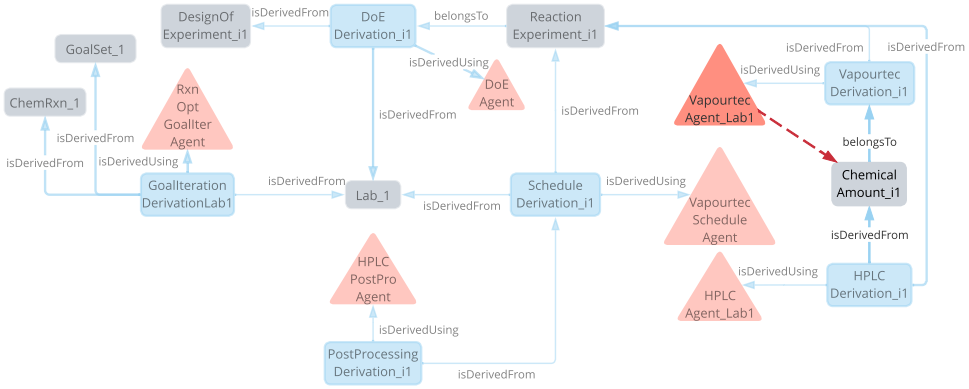
(a) Step 1: Initial workflow requested for the DMTA cycle.



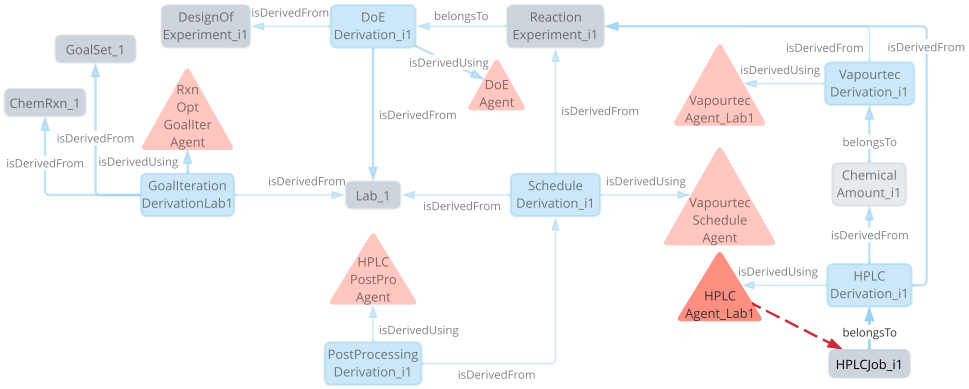
(b) Step 2: New experiment conditions suggested by the DoE Agent.



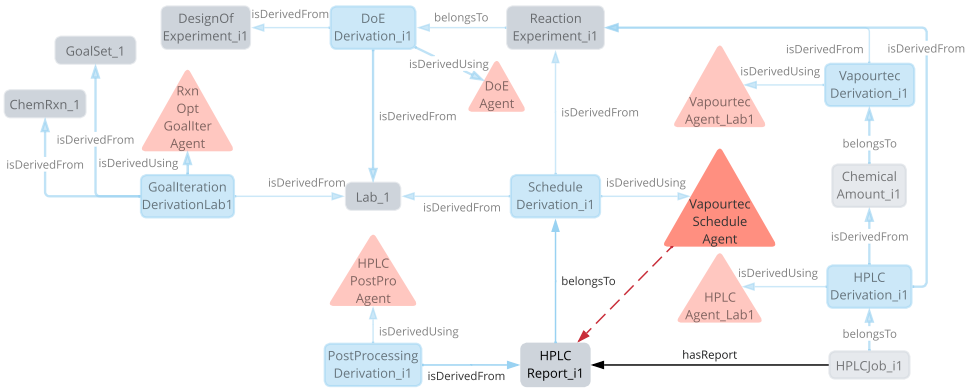
(c) Step 3: Experimentation and characterisation scheduled in the specified laboratory.



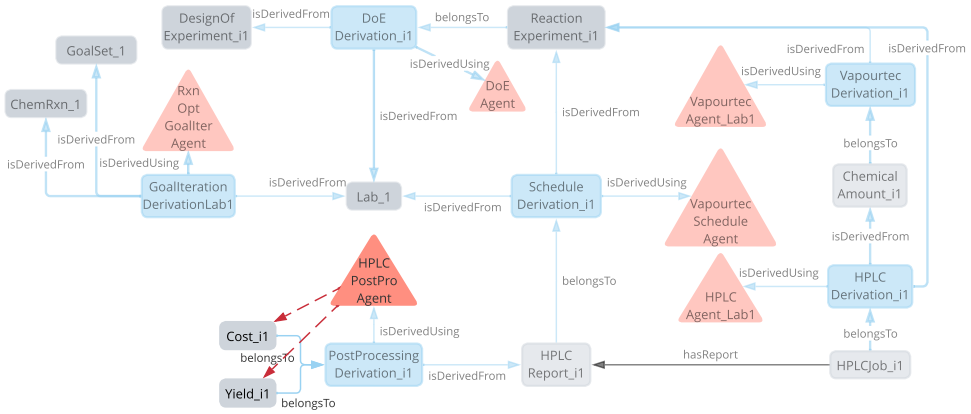
(d) Step 4: Reaction completed and sample injected into HPLC for analysis.



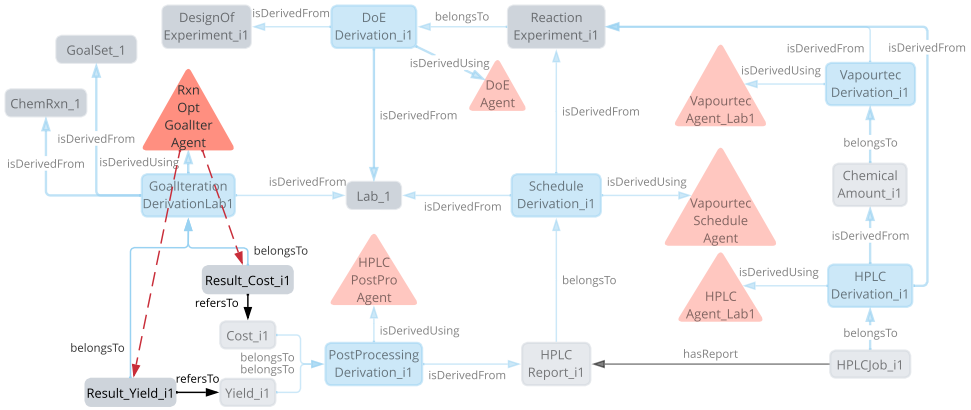
(e) Step 5: HPLC analysis completed with the raw report generated.



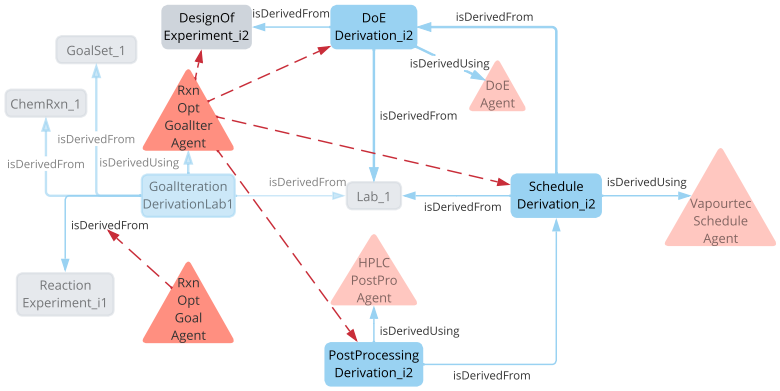
(f) Step 6: HPLC report delegated by the scheduler for post-processing.



(g) Step 7: Performance indicators computed for the reaction experiment.



(h) Step 8: Objectives of interest added as iteration results and returned to decide the next step.



(i) Step 9: Goal iteration entered the next round with the previous experiment as historical data.

Fig. 6.15 Stepping of the derived information generated in the dKG during the initial goal iteration, assuming no prior data. The red dashed lines refer to instantiation by the respective agent. For simplicity, only instances that are necessary to connect the chain are presented. The loop continues until the research goals are met or the resources are used up.

6.3 Results and discussions

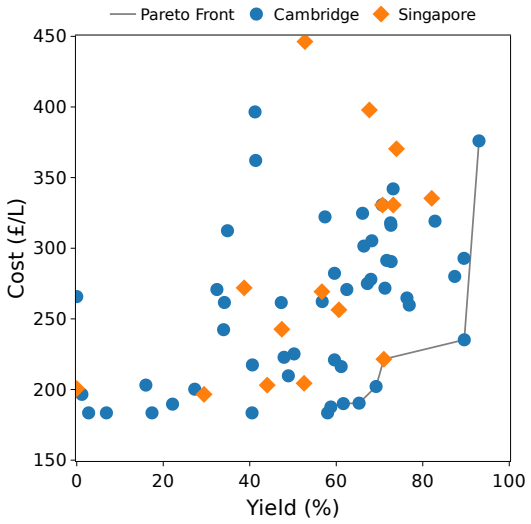
6.3.1 Collaborative closed-loop optimisation

To demonstrate the scalability and modularity, the dKG approach was first applied to a real-time collaborative closed-loop optimisation distributed over two SDLs in Cambridge and Singapore. The objectives selected were run material cost and yield that were sampled for a search space of molar equivalents (relative to benzaldehyde **1**) of acetone **2**, NaOH **3**, residence time and reaction temperature. The research goals and restrictions were populated in the dKG via a web front end. As no prior experimental data was provided, the agents start experiments with random conditions and gradually update their beliefs using TSEMO algorithm [27]. Before running the optimisation, two labs were verified to produce consistent results for two control conditions, in line with the practice of Shields et al. [329]. For experimental details see Appendix C.

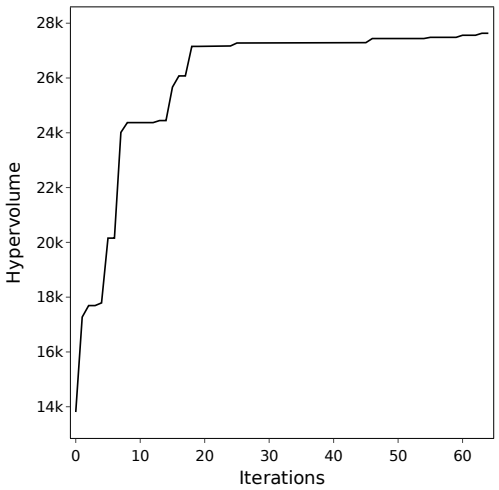
Figure 6.16(a) presents the cost-yield objectives consisting of 65 data points collected during the self-optimisation. Throughout the operation, two SDLs share the results with each other when proposing new experimental conditions. Despite the differences in configurations, the reaction data produced by both machines were interoperable owing to the layered knowledge abstraction. The real-time collaboration demonstrated faster advances in the Pareto front with the highest yield of 93%. The chemicals used in this study were obtained from different vendors compared to Jeraal et al. [178], the cost is therefore not directly comparable due to different prices. Although not considered in the optimisation, the environment factor and space-time yield were found to be highly correlated to the yield objective. The best values obtained are 26.17 and 258.175 g L^{-1} h^{-1} when scaled to the same benzaldehyde injection volume (5 mL), both outperformed the previous study [178].

Figure 6.16(b) presents the hypervolume trajectory computed as a measure for the optimal trade-offs between cost-yield objectives, where a larger hypervolume corresponds to a more optimal Pareto front. Notably, the hypervolume did not show significant improvement after 24 iterations, which is approximately 12 hours since starting the campaign. Therefore, we consider the optimisation campaign has reached the optimal solution for this reaction.

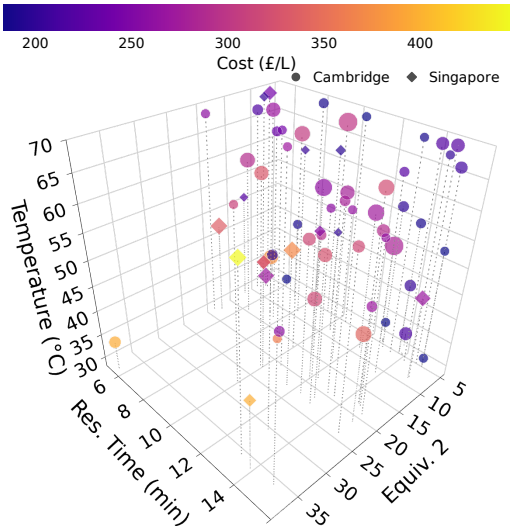
Figures 6.16(c) and 6.16(d) illustrate the influence of the continuous variables on the cost and yield objectives respectively. The cost is calculated to count for the molar amount of input chemicals sourced from the pumps for the reaction. Therefore, it increases linearly with the molar equivalents of the starting materials. Similarly as identified by Jeraal et al. [178], reaction temperature has a positive correlation with the yield of reaction, whereas the residence time shows a poor correlation. Upon examination of the molar equivalent of acetone **2**, it can be observed that its further increase after 30 results in a reduction in yield.



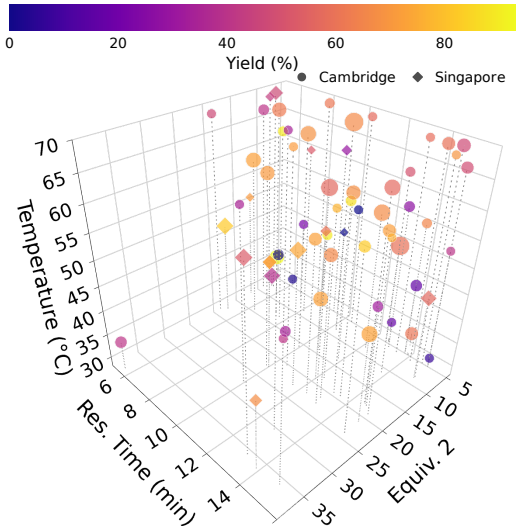
(a) Pareto front plot of the yield and cost objectives for the aldol condensation reaction collaboratively optimised by two distributed SDLs.



(b) Hypervolume trajectory of the optimisation campaign with a reference point of 0% for yield and 500 £ L^{-1} for cost.



(c) Three-dimensional plot of the four sampled design variables colour-coded for run material cost during the closed-loop optimisation. The size of the dots denotes the molar equivalents of 3 in each run.



(d) Three-dimensional plot of the four sampled design variables colour-coded for yield during the closed-loop optimisation. The size of the dots denotes the molar equivalents of 3 in each run.

Fig. 6.16 Objectives and design variables of experiments conducted in the distributed closed-loop optimisation campaign. Each dot refers to a single run.

This decrease can be attributed to the formation of more side product **5** and other further condensation products of acetone and benzaldehyde.

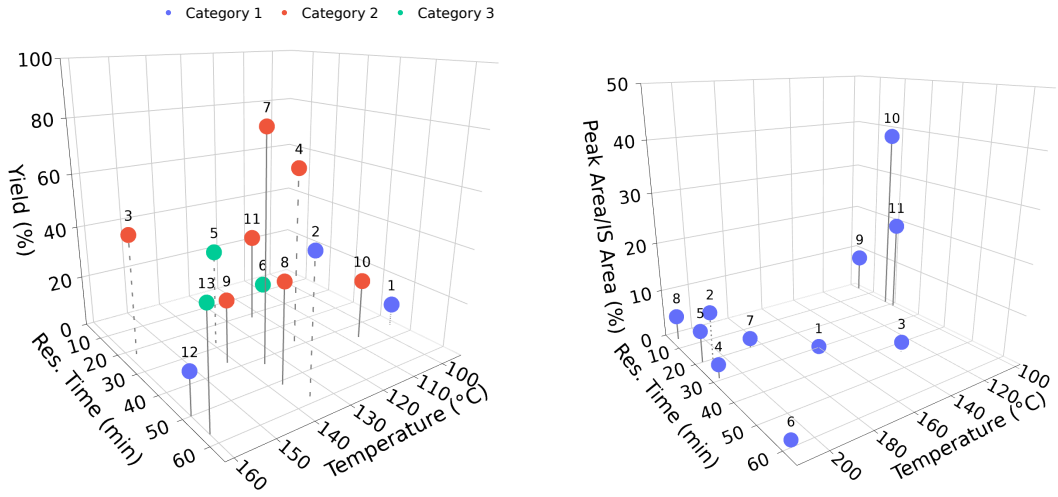
Notably, the Singapore setup encountered an HPLC failure after running for approximately 10 hours. This caused peak shifting of the internal standard which resulted in a wrongly identified peak that gives more than 3500% yield. This point is considered abnormal by the agents and therefore not utilised in the following DoE. An email notification was sent to the developer for maintenance which took the hardware out of the campaign. The asynchronous and distributed design enabled the Cambridge side to further advance the Pareto front for the cost-yield trade-offs. It is also notable that the product peak was missed for one run at the Cambridge side due to a small shift of the peak which gives a yield of 0%. This point was taken into consideration in the DoE, but fortunately, it did not affect the final Pareto front as the corrected yield is still Pareto-dominated. The optimisation campaign was stopped since no more significant improvement was observed in terms of hypervolume, and also due to requests for repurposing the equipment for other projects. The complete provenance records (knowledge graph triples) and interactive animation of the optimisation progress are available in the University of Cambridge data repository (doi:10.17863/CAM.97058).

6.3.2 Efficient synthesis for Suzuki coupling

In the second use case, a two-step Suzuki coupling reaction was chosen to demonstrate the effectiveness of the dKG approach to synthesise challenging molecules. The optimisation was carried out for each step separately due to the limited access to starting materials. The experiments were conducted in Cambridge lab, where Dr Taylor performed all hardware setup and chemical preparations, and Dr Felton developed the Single-Task Bayesian Optimisation (STBO) algorithm. The author set up the dKG following the same procedure as in the distributed SDLs use case. While the details of reactions are in preparation for a journal paper submission and its release is subject to approval by the industry partner, the anonymised reaction results are presented here.

Figure 6.17(a) presents the yield objective consisting of 6 training points and 7 optimisation points for the first step reaction. Two continuous parameters, *i.e.*, residence time (10–60 min) and reaction temperature (90–160 °C), and one categorical parameter, *i.e.*, three combinations of solvent-catalyst-base, were optimised for the maximum product yield output. The molar equivalents of the two reactants and the solvent-catalyst-base combination were kept at 1:1:0.05 for all experiments. One-hot encoding was used to represent the categorical variable as it was proved sufficient to learn from [282]. The liquid handler was used for injecting the chemicals, including the starting materials and catalyst. The digital twin representation enabled the translation of the categorical variable to identify the autosampler

site for the catalyst combination suggested by the algorithm. Following the abstraction of design variables and DoE strategies in OntoDoE, the agent was able to handle categorical and continuous variables at the same time, as well as to use the correct DoE algorithm as specified in the optimisation plan. The framework acts as a "Lego-like" architecture that more DoE algorithms can be easily plugged in. The highest yield was 82% from the catalyst combination category 2.



(a) Three-dimensional plot of the sampled design variables colour-coded for the category of catalyst combination for the first Suzuki coupling reaction.

(b) Three-dimensional plot of the sampled design variables for the second Suzuki coupling reaction.

Fig. 6.17 Objectives and design variables of experiments conducted in the Suzuki coupling use case. Each dot refers to a single run with the annotation indicating the sequence of data acquisition. Points with solid vertical lines refer to the optimisation stage whereas those with dashed lines refer to the training data.

The self-optimisation for the second step was not used as a process optimisation tool *per se* to obtain the maximum yield. Instead, the peak area ratio between the product and the internal standard was chosen as the objective. The idea behind this approach was to access difficult-to-synthesise molecules more easily than if we were to re-design the process routes and spend a significant amount of time trying to synthesise them. Figure 6.17(b) illustrates the objective consisting of 3 points for training and 8 points from self-optimisation of the second step. One of the starting materials for this reaction is the product of the first reaction. The molar equivalents for reactants and the catalyst were kept at 1:1:0.1 for all experiments. Two continuous parameters were optimised for the yield, namely residence time (20–60 min) and reaction temperature (100–210 °C). The lower limit of residence time was adjusted to 5 min after the fourth experiment in the self-optimisation stage, following the chemist's

intuition that a lower residence time could achieve a better area ratio. After the update, the algorithm immediately targeted the new lower bound of residence time, but this resulted in a lower area ratio. Subsequently, the algorithm adapted to vary the reaction temperature and found that a lower temperature would yield a higher area ratio. The self-optimisation campaign was halted due to a shortage of starting material (the product from the first reaction). Nonetheless, we managed to get 82 mg of the final product from a 40 mL scale-up using the optimal condition identified, which is more than enough material for characterisation and bioassays to assess if the active pharmaceutical ingredients should be carried forward into further testing, at which point a route redesign and re-optimisation can be performed.

6.3.3 Lessons learned

The implementation of this work has provided valuable insights and identified areas for future improvement in the realm of dKG systems. In terms of orchestration, it is crucial for the system to be robust to network disruption, especially for the use case that is distributed over the internet. Measures were implemented to ensure that agents deployed in the lab could handle internet cut-offs and resume operations once back online. To minimise downtime during reconnection, future developments could provide on-demand localised deployment of critical parts of the dKG to sustain uninterrupted operation.

For efficient optimisation and data quality, it is critical to have control conditions in place when adding new setups to the network, and only those generated results within the tolerance should be approved. Complex reactions with high-dimensional domains may not be sufficiently evaluated using only two control conditions. This highlights the persisting challenges in maintaining data quality and opens avenues for incorporating strategic cross-workflow validation experiments.

To increase the system's robustness against software and hardware malfunctions, regular backups of all data in the central triplestore should be implemented. Hardware failures during the self-optimisation campaign, which resulted in abnormal data points, also revealed an unresolved issue in automated quality control monitoring. This accentuates the need for a practical solution to bridge the interim technology gap, such as implementing a human-in-the-loop strategy for the effective monitoring of unexpected experimental results.

Further development could also be made to federate the SDLs, where each lab hosts its data and digital twins locally and only exposes its capabilities in the central registry (a "yellow page") without revealing confidential information. An authentication and authorisation mechanism should be added to control access to the equipment and grant permission for federated learning.

The human-machine interaction is also important to accommodate the need for updating the optimisation plan during autonomous experimentation. It would be even more desirable for the SDLs to actively suggest modifications in boundary conditions when they are considered necessary.

When reflecting on the vision of distributed SDLs, our approach exhibits both commonalities and distinctions when compared to contemporary designs. Table 6.1 summarises the key design features, to the best of our knowledge, as they relate to the three major challenges, with the first challenge further divided into the abstraction of resources and workflow coordination.

In terms of resource abstraction, all approaches (including the one presented in this work) employ a modular design that considers hardware limitations in granularity. This modularity is key for a seamless integration of new resources into a plug-and-play system. However, the way resources are exposed to the coordinator varies and this significantly impacts the orchestration of workflows across laboratories. This applies to both workflow template encoding and its actual execution. The dKG approach uses agents acting as lab resource wrappers with KG access. Agents can register for jobs and proactively execute tasks assigned to the digital twin of the resources they manage. This approach is preferable compared to the practices in the remote procedure call paradigm, where lab resources are made accessible as web servers. Based on our experience, it can raise concerns among IT staff when exposing resources across university or company firewalls. Similar to agents, our approach encodes the workflow in the dKG with each step overseen by an agent. Compared to encoding workflows as a sequence of function calls in scripting languages (such as Python), where execution may struggle with asynchronous workflows evolving during optimisation, our approach allows for real-time workflow assembly and modification. For a detailed technical discussion, interested readers can refer to DIF introduced in Chapter 4.

The integration of data serialisation and storage within workflow aims to ease community adoption. As seen in Table 6.1, practices range from transmitting diverse file formats to enforcing a unified data representation. Starting with *ad hoc* ETL tools for new devices prototyping is practical and minimally disruptive when upgrading a single lab. However, we found in Chapter 3 that this approach is less effective for scaling up to a large network of SDLs. This limitation is the driving force behind the development of the dKG approach, despite the initial cost required for creating ontologies that capture a collective understanding of the field. Our design delegates the responsibility of digesting and translating ontologies into the requisite language and file formats to autonomous agents. Compared to adopting a central coordinator to handle data transfer and format translation, our approach emphasises information propagation within a unified data layer, obviating the need for peer-to-peer data

transfer and alleviating network congestion. Drawing an analogy to self-driving cars, once the "driving rules" (ontologies) are learned, SDLs are granted permission to drive on the "road" (information flow). Compared to traditional relational databases used in other studies, where schema modification can be challenging, the open-world assumption inherent in the dKG enhances its extensibility. Organising concepts and relationships within a KG is also more intuitive than traditional tabular structures. However, this flexibility may come at the cost of performance issues when handling extensive data volumes, especially when dealing with data on the scale of ORD. To counter this, technologies such as ontology-based data access [40] can create a virtual KG from relational databases, combining the strengths of both approaches.

Our approach to experimental provenance differs from others due to hardware constraints. It focuses less on exact operation timing, such as robotic arm motions, and more on capturing inputs and outputs within DMTA cycles. This facilitates high-level analysis, enabling answering questions like "which experiments from lab A informed the DoE study for a specific reaction in lab B". This capability has been effectively demonstrated in the interactive Pareto progress animation supporting the distributed SDLs use case (doi:10.17863/CAM.97058). However, for a deeper understanding of epistemic uncertainties associated with operations in complex reactions, it is imperative to expand the ontologies for a more granular abstraction of the experimental procedures. A potential expansion in this regard could involve the ontologisation of χDL.

Looking forward, achieving a globally collaborative research network requires collective efforts. As the dKG aims to reflect a communal understanding of the field, involving different stakeholders early on can accelerate collaboration and increase the chance of success. Specifically, there exists an opportunity for using dKG technology as an integration hub for all aforementioned initiatives. Industrial partners are encouraged to work together and provide a unified API for interacting with their proprietary software and hardware interfaces. This can be facilitated by efforts such as OPC UA [267] and SiLA [313]. Recent studies have shown the successful exchange of HPLC methods between vendors in CDS, demonstrating the potential for the ontology-based approach [281]. Collaboration between scientists and industry is also important at various stages of research and development [52].

Overall, we believe the dKG approach demonstrated in this chapter provides the first evidence of its potential to establish a network of globally distributed SDLs. Although we focus on flow chemistry in this study, the principles are generic. The same approach can be applied to DMTA cycles for other domains should relevant ontologies and agents be made available.

Table 6.1 Comparison between contemporary designs and this work towards the realisation of distributed SDLs.

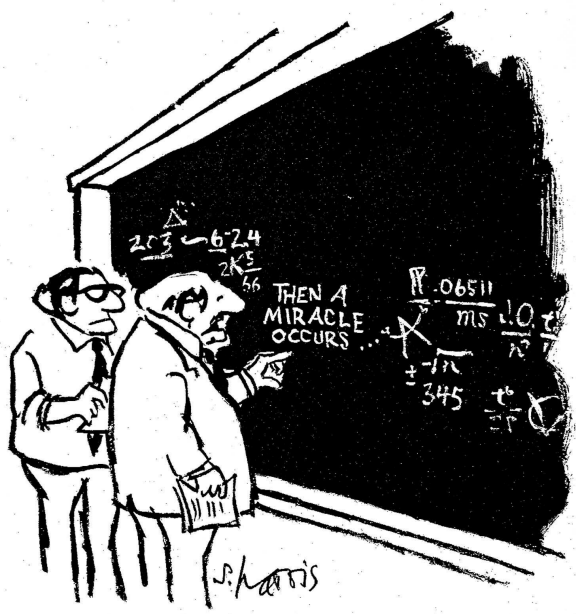
Reference	Resource abstraction	Workflow orchestration	Data serialisation and storage	Experimental provenance
SiLA [313]	Hardware functions are abstracted as SiLA Features following a micro-service architecture with their behaviour described as a state machine.	A sequence of function calls using gRPC and HTTP/2 protocols.	AnIML [313] is employed as a file-less medium for bidirectional analytical data transmission between laboratory information management systems and chromatography data systems.	Device metadata collected during measurements are stored in XML files.
ChemOS [333]	Both software and hardware are abstracted as SiLA servers. Time-consuming computational jobs are managed by AiiDA [168] on SLURM [393].	The central coordinator executes a sequence of Python function calls. The coordinator is also responsible for creating job files in the format required by each hardware/software.	Data are streamed between the central coordinator and each device in diverse file formats, *e.g.*, pickle object, JSON, and CSV. The storage is done in an internal database with a schema consisting of device-agnostic and device-specific tables.	Job execution logs are stored with timestamps in the database table corresponding to each device.
ESCALATE [277]	The Django framework is used to abstract resources as REST API endpoints.	A sequence of steps encoded for "ExperimentTemplate" and accessible via REST API endpoints.	Data are stored in a PostgreSQL database and served via Django REST API [84] endpoints that serialise the data into JSON format for web transfer and inspection.	Metadata associated with the execution of workflow steps are stored with experiment instances in the relational database.
HALEO [289]	Device and their functions ("actions") are represented as hierarchical and asynchronous FastAPI [101] web servers.	A central coordinator executes a sequence of API calls (wrapped as Python functions).	Data are recorded as "groups" and "datasets" in the HDF5 file format and deposited into institutional repositories.	Metadata of "actions" are recorded in the same HDF5 file as the experimental measurements.
χDL [229, 304]	Hardware is categorised/abstracted based on the unit operations in chemical reactions that it can execute.	The sequence of synthesis steps is expressed in XML format, which is later compiled into machine-actionable instructions in a Python script.	The analysis reports are kept in their native file format and bundled with the synthesis description in a PostgreSQL database.	Hardware instructions, the actual performed actions, and other metadata are stored with experiment instances in the relational database.
This work	Hardware is virtualised as a digital twin in a dKG, where its control interface, akin to software resources, is wrapped using the derivation agent template.	DMTA cycles are expressed as directed acyclic graphs of "derivations" in the dKG each referring to a step managed by the respective software agent.	Data are expressed in ontological format (triples) wherever possible. Files (*e.g.*, CSV and XLS) are stored on a file server. Their ontological translation and pointers to the file server location are stored in the triple store.	The inputs/outputs annotation of each step in the workflow is recorded by DIF as triples. The detailed operation timing is not recorded owing to API limitations in obtaining information at this level of granularity.

6.4 Chapter summary

In this chapter, we applied the dKG approach to realise a conceptual architecture for distributed SDLs that integrates data, software, hardware, and workflow into a unified framework. We developed ontologies to represent various aspects of chemical knowledge and hardware digital twins involved in a closed-loop optimisation campaign. By employing autonomous agents as executable knowledge components to update and restructure the dKG, we have enabled collaborative management of data and material flow within and across SDLs. Our approach allows scientists to initiate the autonomous workflow by setting up a goal request, which triggers the flow of information through the dKG as the experimentation workflow progresses. Compared to contemporary designs, where a central orchestrator is responsible for all the data transfer, our approach emphasises information flows in the data layer, reducing the traffic stress on a single point in the network.

Chapter 7

Conclusions



Credit: Sidney Harris ©

"There's just a tremendous amount of craftsmanship in between a great idea and a great product. And as you evolve that great idea, it changes and grows. It never comes out like it starts because you learn a lot more as you get into the subtleties of it."

– Steve Jobs, *The Lost Interview (1995)*

7.1 Summary

The primary goal of this dissertation is to investigate the effectiveness of dKG technology in addressing challenges identified in contemporary SDLs towards a vision of a connected research network. The research outputs from the thesis are summarised below in order to conclude the main contributions of this thesis.

- A review of data infrastructure in SDLs was provided. The analysis identified the current trend in literature as a "platform-based approach" which consists of five functional components. The data format and exchange protocol between these components are reviewed in detail. It was found that the adoption of an *ad hoc* data format and exchange protocol in the selected studies hinders the interoperability between different platforms, which was identified as a key issue to be addressed in contemporary design.

- The review also assessed the ongoing community initiatives related to data standards and exchange protocols for chemistry knowledge and entities involved in the laboratory environment. The analysis revealed a growing interest in employing semantic data representations within the community. When further integrated with autonomous agents, the resulting dKG technology was suggested as a promising solution for addressing the interoperability problem by bringing together the representation of data, software, hardware, and workflow into a unified framework.

- Building on the recommendations from the review, a generic and KG-native infrastructure was developed. The framework introduced the notion of "derivation" to represent both computational and physical experimentation procedures, essentially portraying the process of generating new information from existing inputs. The entity responsible for performing the derivation, such as software or equipment, was encapsulated as an agent. Consequently, workflows were depicted as DAGs of these derivations. Through the ontological representation of data, software, hardware, and workflow, the proposed infrastructure reformulated the scientific discovery process as a dynamic flow of information within the dKG.

- The developed dKG approach was then applied to an automated kinetic mechanism calibration use case executed on an HPC facility. The adoption of ontologies allowed a seamless conversion of different input file formats required by two simulation software. The automated calibration not only achieved significantly more accurate results but also greatly shortened the time required compared to manual calibration. Moreover, this application demonstrated the advantage of dKG in the context of data sharing; for

instance, the computational process linked to IRIs of experimental data, effectively traced the impact of experimental observations on the resulting models. This approach can be viewed as a means of encouraging data citations that incentivise the reporting of "negative results", such as conditions resulting in lower yield, to aid in developing better predictive models.

- Finally, the dKG approach was applied in closed-loop optimisation for wet-lab experiments. The framework was applied to two use cases. The first use case connected two SDLs distributed in Cambridge and Singapore to achieve a collaborative multi-objective self-optimisation. In the second use case, the framework demonstrated its capability to optimise both categorical and continuous variables simultaneously. For both use cases, all data and workflow steps were recorded in the dKG following the FAIR principles. These proof-of-concept studies not only expanded the spectrum of technical choices accessible to the community but also offered valuable insights into guiding the future research directions of the community involved in the development of interoperable SDLs.

7.2 Suggestions for future work

Following the proof-of-concept in this thesis, many future directions exist. Four main directions are proposed below – the topic of which varies in scope, from immediate build-up of the current infrastructure gradually to long-term topics geared toward the development of artificial general intelligence for a sustainable future.

- **Robust infrastructure for multi-cloud and multi-laboratory scientific research.**

  Immediately following the proof-of-concept presented in this work, the key to success lies in the easy adoption of the proposed infrastructure by different groups and organisations. There are four interconnected opportunities within the realm of dKG that can further enhance interoperability with existing initiatives, namely: (1) data: incorporating community-driven public databases, such as ORD [184], and ELNs, such as Chemotion [363]; (2) software: enabling compatibility and interoperability between existing orchestrators such as ChemOS [303] and HELAO [136]; (3) hardware: developing ontologies for commercial and industrial-grade standards of hardware interaction, *e.g.*, SiLA and OPC UA; and (4) workflow: proposing ontologies for data standards that encode synthesis steps and experimental workflows, for instance, χDL [229] and Autoprotocol [234].

These opportunities primarily entail what can be categorised as "engineering challenges" and their effective resolution necessitates a rapid prototyping [11, 217] and close collaboration among all stakeholders within the field. Rather than each group independently developing their own SDL, it will be more beneficial for the community to establish a shared knowledge repository and actively involve all contributions. One potential business model could involve the government funding bodies incentivising researchers to create structured and machine-readable data standards. Hardware vendors should adopt these standards to ensure the data is "born digital". Publishers could also play a role in enforcing data format requirements upon paper acceptance. At the same time, it is important that these standards and resulting data remain in the public domain and allow for version-controlled updates and modifications. This collaborative approach not only promotes efficiency but also fosters a more cohesive and standardised ecosystem within the field, ultimately benefiting the entire community.

- **Human-in-the-loop interaction for scientists and the SDLs.**

  When revisiting the re-grouped structure of MAPs as discussed in Chapter 3, the "receptionist" for handling human inputs emerges as the final piece in the SDLs puzzle. Contrary to the scepticism that automation in chemistry might supplant the traditional bench chemist [31], we believe the advancement of SDLs would enhance the capability of human researchers. Much like the transformative impact of computer and computation on our work productivity, an analogy of the "chemputer/chemputation" has been recurrently explored [143]. This analogy suggests that the progression of SDLs will empower scientists to engage in more creative tasks. However, drawing from our experiences as elucidated in Chapter 6, it is clear that developing an improved human-SDLs interaction mechanism is critical for navigating through complex situations in scientific discovery from a user perspective. The significance of such an approach has been exemplified by the success of ChatGPT/GPT-4 developed by OpenAI [269].

  In a goal-oriented SDL, human involvement is required from two aspects, including (re)defining goals and diagnosing/recovering from faulty states. The process of setting goals for SDLs is conceptually analogous to determining destinations for self-driving cars, and it often involves discussions considering boundary conditions. For instance, researchers may have preferences regarding synthetic routes or may wish to adjust objective priorities as experiments unfold. In the Suzuki coupling use case discussed in Chapter 6, the alternation of the boundary conditions was driven by the intuition of chemists, who made these adjustments based on their continuous monitoring of email notifications related to the experimental progress. However, it would be ad-

vantageous if the SDLs could autonomously propose modifications to scientists in situations like these. This would not only enhance the experiments' efficiency but also prove especially beneficial for graduate students who are still in the learning phase, providing them with valuable guidance and enhancing their overall learning experience. On the fault recovery side, the intricate and interdisciplinary nature of SDLs typically requires a comprehensive system overview to swiftly identify and rectify issues. The current design implements these communications in a for-your-information style, with researchers receiving automated notifications of experimental progress and hardware failure via no-reply emails. Subsequent manual modifications to the dKG are often necessary after issue resolution. Ideally, SDLs should facilitate interactive conversations for diagnosing the root causes of issues. This capability would prove particularly beneficial for researchers who may not have been involved in the initial development of the SDLs. Achieving this level of versatility necessitates the SDLs to be programmable through both text and voice inputs. Early instances of this direction, such as using AI as a co-pilot for experimentalists [296], offer promise, although such capabilities within the dKG domain remain relatively underexplored.

- **Combination of dKG and Large Language Models (LLMs) in scientific research.**

  The chemistry and materials domain involves a vast expanse of knowledge that necessitates an efficient way for the holistic integration of SDLs. In this regard, LLMs present a promising solution due to their substantial repository of general knowledge. Previous studies have demonstrated the utility of LLMs in chemical research, including synthesis planning and execution [29], text-mining synthesis conditions [398], and translating synthesis steps from natural language to chemical description (χDL) [336]. However, it is important to note that these studies have also acknowledged the well-known issue of LLM hallucination. Caution is therefore warranted when deploying LLMs in applications with potential real-world consequences. In the context of SDLs or any scenario involving interactions with the physical world, a noteworthy limitation of LLMs is their restricted comprehension of the tangible (physical) environment [388]. To address this gap, integrating LLMs with dKG appears to be a compelling proposition.

  One potential opportunity pertains to the utilisation of LLMs to autonomously generate drivers for new laboratory equipment based on their existing documentation and knowledge stored in dKG. It aims to alleviate the challenge of coding drivers for diverse hardware, especially legacy equipment that may not be practical to obtain from individual hardware vendors [274]. One follow-up opportunity is the development of LLM-based agents to transition from automation to autonomy in SDLs' workflows.

These agents can learn from previous dKG recordings and formulate workflows using the newly integrated software/hardware. In this paradigm, LLMs are not only users of tools [285] but also their creators [39].This modularity resonates well with the concept of *chain-of-thought* [379], which has shown promise in augmenting the cognitive capacities of LLMs. Furthermore, it aligns with DIF presented in Chapter 4, where information dynamically flows in the dKG through the actions of agents. LLMs could thus be integrated with DIF to address questions about where and how something originates from. The dKG would not only function as a comprehensive world model for LLMs but also offer crucial endpoints for fact-checking, ultimately enhancing the reliability and autonomy of SDLs.

- **Hierarchical goal alignment for autonomous agents with Sustainable Development Goals (SDGs).**

  Chapter 6 briefly touched upon the concept of goal derivation, a process of breaking down overarching goals into smaller sub-goals for various autonomous agents. This concept has far greater influence than only a lab experiment. In fact, it aligns well with how humans behave [236]. Envisioning a world quantified by real-time sensor data, populated by numerous autonomous software agents pursuing their unique goals, the alignment of these objectives with the SDGs becomes paramount. This also echoes the previous points on the interaction between humans and autonomous agents. DIF developed in Chapter 4 plays a vital role in enabling this vision as it realises the mechanism of agents behaving autonomously toward their goals.

  Expanding upon the realm of SDLs, the linked nature of the semantic web offers opportunities to extend this network to smart factories, smart buildings, and smart grids [310]. Demonstrations of this concept are already visible in applications like the World Avatar for smart city planning [45] and the UK national digital twin [3]. By connecting SDLs with this broader context, we anticipate facilitating multi-scale and cross-domain interactions among scientists, engineers, and policymakers to investigate the impact of laboratory research on the global landscape and to shape the direction of scientific advancement by backcasting from the SDGs.

The vision of establishing a global collaborative research network holds the potential to foster interdisciplinary studies and accelerate the realisation of a sustainable society. As we advance towards this ambitious vision, our understanding of dKG technology is likely to refine and broaden. Echoing Steve Jobs' comments, this path will be characterised by continuous growth and deeper insights into its nuances. We eagerly anticipate this "miracle" to occur in the near future.

References

[1] M. Abolhasani and E. Kumacheva. The Rise of Self-Driving Labs in Chemical and Materials Sciences. *Nat. Synth.*, 2(6):483–492, 2023. doi:10.1038/s44160-022-00231-0.

[2] Air Force Research Laboratory. ARES OS, 2021. URL https://github.com/AFRL-ARES/ARES_OS. Accessed 13 November 2021.

[3] J. Akroyd, S. Mosbach, A. Bhave, and M. Kraft. Universal Digital Twin – A Dynamic Knowledge Graph. *Data-Centric Eng.*, 2:e14, 2021. doi:10.1017/dce.2021.10.

[4] J. Akroyd, A. Bhave, G. Brownbridge, E. Christou, M. Hillman, M. Hofmeister, M. Kraft, J. Lai, K. F. Lee, S. Mosbach, D. Nurkowski, and O. Parry. CReDo Technical Paper 1: Building a Cross-Sector Digital Twin, 2022. URL https://digitaltwinhub.co.uk/credo/technical/1-building-a-cross-sector-twin/. Accessed 19 July 2022.

[5] J. Akroyd, Z. Harper, D. Soutar, F. Farazi, A. Bhave, S. Mosbach, and M. Kraft. Universal Digital Twin: Land Use. *Data-Centric Eng.*, 3, 2022. doi:10.1017/dce.2021.21.

[6] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock. Kepler: An Extensible System for Design and Execution of Scientific Workflows. In *Proceedings of 16th International Conference on Scientific and Statistical Database Management, 2004.*, pages 423–424. IEEE, 2004. doi:10.1109/SSDM.2004.1311241.

[7] AnIML Working Group. AnIML: Overview, 2021. URL https://www.animl.org/overview. Accessed 31 July 2021.

[8] A. Aspuru-Guzik and K. Persson. Materials Acceleration Platform: Accelerating Advanced Energy Materials Discovery by Integrating High-Throughput Methods and Artificial Intelligence. Mission Innovation: Innovation Challenge 6, 2018. URL http://nrs.harvard.edu/urn-3:HUL.InstRepos:35164974. Accessed 12 November 2021.

[9] Atinary Technologies Inc. Enabling Self-Driving Labs® Technology, 2023. URL https://atinary.com/. Accessed 17 July 2023.

[10] P. Azadi, G. Brownbridge, I. Kemp, S. Mosbach, J. S. Dennis, and M. Kraft. Microkinetic Modeling of the Fischer-Tropsch Synthesis over Cobalt Catalysts. *ChemCatChem*, 7(1):137–143, 2015. doi:10.1002/cctc.201402662.

[11] S. G. Baird and T. D. Sparks. What is a Minimal Working Example for a Self-Driving Laboratory? *Matter*, 5(12):4170–4178, 2022. doi:10.1016/j.matt.2022.11.007.

[12] M. Baker. 1, 500 Scientists Lift the Lid on Reproducibility. *Nature*, 533(7604): 452–454, 2016. doi:10.1038/533452a.

[13] J. B. L. Bard and S. Y. Rhee. Ontologies in Biology: Design, Applications and Future Challenges. *Nat. Rev. Genet.*, 5(3):213–222, 2004. doi:10.1038/nrg1295.

[14] J. Barrasa. RDF Triple Stores vs. Labeled Property Graphs: What's the Difference?, 2017. URL https://neo4j.com/blog/rdf-triple-store-vs-labeled-property-graph-difference/. Accessed 17 September 2023.

[15] C. Batchelor and P. Corbett. Semantic Enrichment of Journal Articles Using Chemical Named Entity Recognition. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 45–48, 2007. URL https://aclanthology.org/P07-2012. Accessed 31 July 2021.

[16] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, and G. Carothers. RDF 1.1 Turtle – Terse RDF Triple Language. W3C Recommendation 25 February 2014, 2014. URL https://www.w3.org/TR/turtle/. Accessed 15 September 2023.

[17] A.-C. Bédard, A. Adamo, K. C. Aroh, M. G. Russell, A. A. Bedermann, J. Torosian, B. Yue, K. F. Jensen, and T. F. Jamison. Reconfigurable System for Automated Optimization of Diverse Chemical Reactions. *Science*, 361(6408):1220–1225, 2018. doi:10.1126/science.aat0650.

[18] T. Berners-Lee. World-Wide Computer. *Commun. ACM*, 40(2):57–58, 1997. doi:10.1145/253671.253704.

[19] T. Berners-Lee. Linked Data, 2006. URL https://www.w3.org/DesignIssues/LinkedData.html. Accessed 15 September 2023.

[20] T. Berners-Lee. Is Your Linked Open Data 5 Star?, 2010. URL http://www.w3.org/DesignIssues/LinkedData#fivestar. Accessed 15 September 2023.

[21] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Sci. Am.*, 284(5): 34–43, 2001. doi:10.1038/scientificamerican0501-34.

[22] P. L. Bhoorasingh, B. L. Slakman, F. Seyedzadeh Khanshan, J. Y. Cain, and R. H. West. Automated Transition State Theory Calculations for High-Throughput Kinetics. *J. Phys. Chem. A*, 121(37):6896–6904, 2017. doi:10.1021/acs.jpca.7b07361.

[23] G. Blanke. The Unified Data Model (UDM). NIH Virtual Workshop on Reaction Informatics, May 18-20, 2021. URL https://cactus.nci.nih.gov/presentations/NIHReactInf_2021-05/UDM_at_NIH_Reaction_conference_May_2021_-_Gerd_Blanke.pdf. Accessed 13 November 2021.

[24] blazegraph. Reification Done Right, 2020. URL https://github.com/blazegraph/database/wiki/Reification_Done_Right. Accessed 29 September 2023.

[25] G. Blondet, J. Le Duigou, N. Boudaoud, and B. Eynard. An Ontology for Numerical Design of Experiments Processes. *Comput. Ind.*, 94:26–40, 2018. doi:10.1016/j.compind.2017.09.005.

[26] R. H. Bordini, L. Braubach, M. Dastani, A. E. F. Seghrouchni, J. J. Gomez-Sanz, J. Leite, G. O'Hare, A. Pokahr, and A. Ricci. A Survey of Programming Languages and Platforms for Multi-Agent Systems. *Informatica (Ljubljana)*, 30(1):33–44, 2006. URL https://www.informatica.si/index.php/informatica/article/view/71.

[27] E. Bradford, A. M. Schweidtmann, and A. Lapkin. Efficient Multiobjective Optimization Employing Gaussian Processes, Spectral Sampling and A Genetic Algorithm. *J. Glob. Optim.*, 71(2):407–438, 2018. doi:10.1007/s10898-018-0609-2.

[28] J. Bradshaw, M. J. Kusner, B. Paige, M. H. S. Segler, and J. M. Hernández-Lobato. A Generative Model for Electron Paths. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*, pages 1–19, 2019. URL https://arxiv.org/abs/1805.10970.

[29] A. M. Bran, S. Cox, A. D. White, and P. Schwaller. ChemCrow: Augmenting Large-Language Models with Chemistry Tools. *arXiv Preprint*, 2023. doi:10.48550/arXiv.2304.05376.

[30] M. Bratman. *Intention, Plans, and Practical Reason*. Cambridge: Cambridge, MA: Harvard University Press, 1987.

[31] R. Brazil. Automation in the Chemistry Lab., 2021. URL https://www.chemistryworld.com/careers/automation-in-the-chemistry-lab/4012832.article. Accessed 31 July 2021.

[32] C. P. Breen, A. M. K. Nambiar, T. F. Jamison, and K. F. Jensen. Ready, Set, Flow! Automated Continuous Synthesis and Optimization. *Trends Chem.*, 3(5):373–386, 2021. doi:10.1016/j.trechm.2021.02.005.

[33] A. Brendel, F. Dorfmüller, A. Liebscher, P. Kraus, K. Kress, H. Oehme, M. Arnold, and R. Koschitzki. Laboratory and Analytical Device Standard (LADS): A Communication Standard Based on OPC UA for Networked Laboratories. In *Smart Biolabs of the Future*, pages 175–194. Springer International Publishing, 2022. doi:10.1007/10_2022_209.

[34] D. Bryce, R. P. Goldman, M. DeHaven, J. Beal, B. Bartley, T. T. Nguyen, N. Walczak, M. Weston, G. Zheng, J. Nowak, P. Lee, J. Stubbs, N. Gaffney, M. W. Vaughn, C. J. Myers, R. C. Moseley, S. Haase, A. Deckard, B. Cummins, and N. Leiby. Round Trip: An Automated Pipeline for Experimental Design, Execution, and Analysis. *ACS Synth. Biol.*, 11(2):608–622, 2022. doi:10.1021/acssynbio.1c00305.

[35] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, N. Rankin, B. Harris, R. S. Sprick, and A. I. Cooper. A Mobile Robotic Chemist. *Nature*, 583(7815):237–241, 2020. doi:10.1038/s41586-020-2442-2.

[36] S. M. Burke, U. Burke, R. Mc Donagh, O. Mathieu, I. Osorio, C. Keesee, A. Morones, E. L. Petersen, W. Wang, T. A. DeVerter, M. A. Oehlschlaeger, B. Rhodes, R. K. Hanson, D. F. Davidson, B. W. Weber, C.-J. Sung, J. Santner, Y. Ju, F. M. Haas, F. L. Dryer, E. N. Volkov, E. J. K. Nilsson, A. A. Konnov, M. Alrefae, F. Khaled, A. Farooq, P. Dirrenberger, P.-A. Glaude, F. Battin-Leclerc, and H. J. Currana. An

Experimental and Modeling Study of Propene Oxidation. Part 2: Ignition Delay Time and Flame Speed Measurements. *Combust. Flame*, 162(2):296–314, 2015. doi:10.1016/j.combustflame.2014.07.032.

[37] L. Cai and H. Pitsch. Mechanism Optimization Based on Reaction Rate Rules. *Combust. Flame*, 161(2):405–415, 2014. doi:10.1016/j.combustflame.2013.08.024.

[38] L. Cai, S. Jacobs, R. Langer, F. vom Lehn, K. A. Heufer, and H. Pitsch. Autoignition of Oxymethylene Ethers (OMMn, n = 2–4) as Promising Synthetic E-fuels from Renewable Electricity: Shock Tube Experiments and Automatic Mechanism Generation. *Fuel*, 264:116711, 2020. doi:10.1016/j.fuel.2019.116711.

[39] T. Cai, X. Wang, T. Ma, X. Chen, and D. Zhou. Large Language Models as Tool Makers. *arXiv Preprint*, 2023. doi:10.48550/arXiv.2305.17126.

[40] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, and G. Xiao. Ontop: Answering SPARQL Queries over Relational Databases. *Semant. Web*, 8(3):471–487, 2016. doi:10.3233/sw-160217.

[41] L. Cao, D. Russo, K. Felton, D. Salley, A. Sharma, G. Keenan, W. Mauer, H. Gao, L. Cronin, and A. A. Lapkin. Optimization of Formulations Using Robotic Experiments Driven by Machine Learning DoE. *Cell Rep. Phys. Sci.*, 2(1):100295, 2021. doi:10.1016/j.xcrp.2020.100295.

[42] L. Cao, D. Russo, and A. A. Lapkin. Automated Robotic Platforms in Design and Development of Formulations. *AIChE J.*, 67(5):e17248, 2021. doi:10.1002/aic.17248.

[43] D. Caramelli, D. Salley, A. Henson, G. A. Camarasa, S. Sharabi, G. Keenan, and L. Cronin. Networking Chemical Robots for Reaction Multitasking. *Nat. Commun.*, 9:3406, 2018. doi:10.1038/s41467-018-05828-8.

[44] Carnegie Mellon University and Emerald Cloud Lab. CMU Cloud Lab, 2023. URL https://cloudlab.cmu.edu/. Accessed 12 May 2023.

[45] A. Chadzynski, N. Krdzavac, F. Farazi, M. Q. Lim, S. Li, A. Grisiute, P. Herthogs, A. von Richthofen, S. Cairns, and M. Kraft. Semantic 3D City Database - An Enabler for a Dynamic Geospatial Knowledge Graph. *Energy and AI*, 6:100106, 2021. doi:10.1016/j.egyai.2021.100106.

[46] Y. Chang, M. Jia, Y. Liu, Y. Li, and M. Xie. Development of a New Skeletal Mechanism for n-Decane Oxidation under Engine-relevant Conditions Based on a Decoupling Methodology. *Combust. Flame*, 160(8):1315–1332, 2013. doi:10.1016/j.combustflame.2013.02.017.

[47] R. Chao, Y. Yuan, and H. Zhao. Building Biological Foundries for Next-Generation Synthetic Biology. *Sci. China: Life Sci.*, 58(7):658–665, 2015. doi:10.1007/s11427-015-4866-8.

[48] R. Chard, Z. Li, K. Chard, L. Ward, Y. Babuji, A. Woodard, S. Tuecke, B. Blaiszik, M. J. Franklin, and I. Foster. DLHub: Model and Data Serving for Science. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 283–292. IEEE, 2019. URL https://arxiv.org/abs/1811.11213.

[49] R. Chard, J. Pruyne, K. McKee, J. Bryan, B. Raumann, R. Ananthakrishnan, K. Chard, and I. T. Foster. Globus Automation Services: Research Process Automation across the Space–Time Continuum. *Future Gener. Comput. Syst.*, 142:393–409, 2023. doi:10.1016/j.future.2023.01.010.

[50] S. Chatterjee, M. Guidi, P. H. Seeberger, and K. Gilmore. Automated Radial Synthesis of Organic Molecules. *Nature*, 579(7799):379–384, 2020. doi:10.1038/s41586-020-2083-5.

[51] Chemical Semantics. GNVC: Gainesville Core Ontology - Standard for Publishing Results of Computational Chemistry, 2015. URL http://ontologies.makolab.com/gc/gc07.owl. Accessed 21 September 2021.

[52] M. Christensen, L. P. Yunker, P. Shiri, T. Zepel, P. L. Prieto, S. Grunert, F. Bork, and J. E. Hein. Automation isn't Automatic. *Chem. Sci.*, 12(47):15473–15490, 2021. doi:10.1039/D1SC04588A.

[53] M. Christensen, L. P. E. Yunker, F. Adedeji, F. Häse, L. M. Roch, T. Gensch, G. dos Passos Gomes, T. Zepel, M. S. Sigman, A. Aspuru-Guzik, and J. Hein. Data-Science Driven Autonomous Process Optimization. *Commun. Chem.*, 4:112, 2021. doi:10.1038/s42004-021-00550-x.

[54] P. Ciccarese, S. Soiland-Reyes, K. Belhajjame, A. J. G. Gray, C. Goble, and T. Clark. PAV Ontology: Provenance, Authoring and Versioning. *J. Biomed. Semant.*, 4(1):37, 2013. doi:10.1186/2041-1480-4-37. URL https://pav-ontology.github.io/pav/.

[55] A. D. Clayton, J. A. Manson, C. J. Taylor, T. W. Chamberlain, B. A. Taylor, G. Clemens, and R. A. Bourne. Algorithms for the Self-Optimisation of Chemical Reactions. *React. Chem. Eng.*, 4(9):1545–1554, 2019. doi:10.1039/C9RE00209J.

[56] CMCL Innovations. *k*inetics & SRM engine suite (version 2020.1.1), 2020. URL https://cmclinnovations.com/solutions/products/kinetics/. Accessed 6 March 2021.

[57] CMCL Innovations. MoDS: Model Development Suite (version 2020.2.2), 2020. URL https://cmclinnovations.com/solutions/products/mods/. Accessed 6 March 2021.

[58] S. J. Coles, N. E. Day, P. Murray-Rust, H. S. Rzepa, and Y. Zhang. Enhancement of the Chemical Semantic Web through the Use of InChI Identifiers. *Org. Biomol. Chem.*, 3(10):1832–1834, 2005. doi:10.1039/B502828K.

[59] C. W. Coley, D. A. Thomas, J. A. M. Lummiss, J. N. Jaworski, C. P. Breen, V. Schultz, T. Hart, J. S. Fishman, L. Rogers, H. Gao, R. W. Hicklin, P. P. Plehiers, J. Byington, J. S. Piotti, W. H. Green, A. J. Hart, T. F. Jamison, and K. F. Jensen. A Robotic Platform for Flow Synthesis of Organic Compounds Informed by AI Planning. *Science*, 365 (6453):eaax1566, 2019. doi:10.1126/science.aax1566.

[60] C. W. Coley, N. S. Eyke, and K. F. Jensen. Autonomous Discovery in the Chemical Sciences Part I: Progress. *Angew. Chem., Int. Ed.*, 59(51):22858–22893, 2020. doi:10.1002/anie.201909987.

[61] C. W. Coley, N. S. Eyke, and K. F. Jensen. Autonomous Discovery in the Chemical Sciences Part II: Outlook. *Angew. Chem., Int. Ed.*, 59(52):23414–23436, 2020. doi:10.1002/anie.201909989.

[62] E. J. Corey. General Methods for the Construction of Complex Molecules. *Pure Appl. Chem.*, 14(1):19–38, 1967. doi:10.1351/pac196714010019.

[63] E. J. Corey and W. T. Wipke. Computer-Assisted Design of Complex Organic Syntheses. *Science*, 166(3902):178–192, 1969. doi:10.1126/science.166.3902.178.

[64] S. Cox, C. Little, J. R. Hobbs, and F. Pan. Time Ontology in OWL. W3C Candidate Recommendation 26 March 2020, 2020. URL https://www.w3.org/TR/owl-time/. Accessed 21 July 2022.

[65] R. F. da Silva, H. Casanova, K. Chard, D. Laney, D. Ahn, S. Jha, C. Goble, L. Ramakrishnan, L. Peterson, B. Enders, D. Thain, I. Altintas, Y. Babuji, R. M. Badia, V. Bonazzi, T. Coleman, M. Crusoe, E. Deelman, F. D. Natale, P. D. Tommaso, T. Fahringer, R. Filgueira, G. Fursin, A. Ganose, B. Gruning, D. S. Katz, O. Kuchar, A. Kupresanin, B. Ludascher, K. Maheshwari, M. Mattoso, K. Mehta, T. Munson, J. Ozik, T. Peterka, L. Pottier, T. Randles, S. Soiland-Reyes, B. Tovar, M. Turilli, T. Uram, K. Vahi, M. Wilde, M. Wolf, and J. Wozniak. Workflows Community Summit: Bringing the Scientific Workflows Community Together. *arXiv Preprint*, 2021. doi:10.48550/arXiv.2103.09181.

[66] L. Daniele, R. Garcia-Castro, M. Lefrançois, and M. Poveda-Villalon. SAREF: The Smart Applications REFerence ontology, 2020. URL https://saref.etsi.org/core/v3.1.1/. Accessed 21 February 2023.

[67] Dapr Authors. APIs for Building Portable and Reliable Microservices, 2022. URL https://dapr.io/. Accessed 14 July 2022.

[68] Daylight. SMARTS - A Language for Describing Molecular Patterns, 2014. URL https://www.daylight.com/dayhtml/doc/theory/theory.smarts.html. Accessed 27 May 2021.

[69] Daylight. SMIRKS - A Reaction Transform Language, 2014. URL https://www.daylight.com/dayhtml/doc/theory/theory.smirks.html. Accessed 27 May 2021.

[70] DCMI Usage Board. Bibliographic Ontology (BIBO) in RDF, 2016. URL https://www.dublincore.org/specifications/bibo/bibo/. Accessed 29 September 2023.

[71] DCMI Usage Board. DCMI Metadata Terms, 2020. URL https://www.dublincore.org/specifications/dublin-core/dcmi-terms/. Accessed 29 September 2023.

[72] L. De Silva, F. R. Meneguzzi, and B. Logan. BDI Agent Architectures: A Survey. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI), 2020, Japão.*, 2020. doi:10.24963/ijcai.2020/684.

[73] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good, A. Laity, J. C. Jacob, and D. S. Katz. Pegasus: A Framework for Mapping Complex Scientific Workflows onto Distributed Systems. *Sci. Program.*, 13(3):219–237, 2005. doi:10.1155/2005/128026.

[74] E. Deelman, D. Gannon, M. Shields, and I. Taylor. Workflows and e-Science: An Overview of Workflow System Features and Capabilities. *Future Gener. Comput. Syst.*, 25(5):528–540, 2009. doi:10.1016/j.future.2008.06.012.

[75] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. F. Da Silva, M. Livny, and K. Wenger. Pegasus, A Workflow Management System for Science Automation. *Future Gener. Comput. Syst.*, 46:17–35, 2015. doi:10.1016/j.future.2014.10.008.

[76] E. Deelman, T. Peterka, I. Altintas, C. D. Carothers, K. K. van Dam, K. Moreland, M. Parashar, L. Ramakrishnan, M. Taufer, and J. Vetter. The Future of Scientific Workflows. *J. High Perform. Comput. Appl.*, 32(1):159–175, 2018. doi:10.1177/1094342017704893.

[77] F. Delgado-Licona and M. Abolhasani. Research Acceleration in Self-Driving Labs: Technological Roadmap toward Accelerated Materials and Molecular Discovery. *Adv. Intell. Syst.*, 5(4):2200331, 2022. doi:10.1002/aisy.202200331.

[78] W. P. Dempsey, I. Foster, S. Fraser, and C. Kesselman. Sharing Begins at Home: How Continuous and Ubiquitous FAIRness Can Enhance Research Productivity and Data Reuse. *Harvard Data Sci. Rev.*, 2022. doi:10.1162/99608f92.44d21b86.

[79] J. R. Deneault, J. Chang, J. Myung, D. Hooper, A. Armstrong, M. Pitt, and B. Maruyama. Toward Autonomous Additive Manufacturing: Bayesian Optimization on a 3D Printer. *MRS Bull.*, 46:566–575, 2021. doi:10.1557/s43577-021-00051-1.

[80] Department for Environment Food & Rural Affairs. Real Time flood-monitoring API, 2021. URL https://environment.data.gov.uk/flood-monitoring/doc/reference. Accessed 4 February 2022.

[81] Department for Levelling Up, Housing & Communities. Energy Performance of Buildings Data, 2022. URL https://epc.opendatacommunities.org/docs/api. Accessed 24 February 2022.

[82] P. Di Tommaso, M. Chatzou, E. W. Floden, P. P. Barja, E. Palumbo, and C. Notredame. Nextflow Enables Reproducible Computational Workflows. *Nat. Biotechnol.*, 35(4): 316–319, 2017. doi:10.1038/nbt.3820.

[83] T. Dimitrov, C. Kreisbeck, J. S. Becker, A. Aspuru-Guzik, and S. K. Saikin. Autonomous Molecular Design: Then and Now. *ACS Appl. Mater. Interfaces*, 11(28): 24825–24836, 2019. doi:10.1021/acsami.9b01226.

[84] Django Developers. Web APIs for Django, 2023. URL https://github.com/encode/django-rest-framework. Accessed 19 October 2023.

[85] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina. Microservices: Yesterday, Today, and Tomorrow. In *Present and Ulterior Software Engineering*, pages 195–216. Springer International Publishing, 2017. doi:10.1007/978-3-319-67425-4_12.

[86] J. A. Dreyer, M. N. Jones, J. Kager, S. Larsen, J. M. Woodley, K. Dam-Johansen, and J. K. Huusom. Digitalisering af Forskning og Undervisning på DTU Kemiteknik. *Dansk Kemi*, 104(1):6–11, 2023. URL https://www.kemifokus.dk/digitalisering-af-forskning-og-undervisning-paa-dtu-kemiteknik/.

[87] B. DuCharme. *Learning SPARQL: Querying and Updating with SPARQL 1.1*. O'Reilly Media, Inc., 2nd edition, 2013.

[88] A. M. Duffield, A. V. Robertson, C. Djerassi, B. G. Buchanan, G. L. Sutherland, E. A. Feigenbaum, and J. Lederberg. Applications of Artificial Intelligence for Chemical Inference. II. Interpretation of Low-Resolution Mass Spectra of Ketones. *J. Am. Chem. Soc.*, 91(11):2977–2981, 1969. doi:10.1021/ja01039a026.

[89] A. Eibeck, M. Q. Lim, and M. Kraft. J-Park Simulator: An Ontology-Based Platform for Cross-domain Scenarios in Process Industry. *Comput. Chem. Eng.*, 131:106586, 2019. doi:10.1016/j.compchemeng.2019.106586.

[90] A. Eibeck, A. Chadzynski, M. Q. Lim, K. Aditya, L. Ong, A. Devanand, G. Karmakar, S. Mosbach, R. Lau, I. A. Karimi, E. Y. S. Foo, and M. Kraft. A Parallel World Framework for Scenario Analysis in Knowledge Graphs. *Data-Centric Eng.*, 1:e6, 2020. doi:10.1017/dce.2020.6.

[91] S. Elder, C. Klumpp-Thomas, A. Yasgar, J. Travers, S. Frebert, K. M. Wilson, A. V. Zakharov, J. L. Dahlin, C. Kreisbeck, D. Sheberla, G. S. Sittampalam, A. G. Godfrey, A. Simeonov, and S. Michael. Cross-Platform Bayesian Optimization System for Autonomous Biological Assay Development. *SLAS Technol.*, 26(6):579–590, 2021. doi:10.1177/24726303211053782.

[92] EMBL-EBI. Molecular Process Ontology, 2014. URL https://www.ebi.ac.uk/ols/ontologies/mop. Accessed 14 June 2021.

[93] EMBL-EBI. Chemical Methods Ontology, 2019. URL https://www.ebi.ac.uk/ols/ontologies/chmo. Accessed 14 June 2021.

[94] EMBL-EBI. Name Reaction Ontology, 2021. URL https://www.ebi.ac.uk/ols/ontologies/rxno. Accessed 30 May 2023.

[95] ESCALATE. Interacting with the ESCALATE REST API, 2021. URL https://github.com/darkreactions/ESCALATE/blob/master/demonstrations/REST_API_DEMO.ipynb. Accessed 15 November 2021.

[96] H. Fakhruldeen, D. Marquez-Gamez, and A. I. Cooper. Development of a ROS Driver and Support Stack for the KMR iiwa Mobile Manipulator. In *Annual Conference Towards Autonomous Robotic Systems*, pages 304–314. Springer, 2021. doi:10.1007/978-3-030-89177-0_31.

[97] W. Falcon and The PyTorch Lightning team. PyTorch Lightning, 3 2019. URL https://github.com/Lightning-AI/lightning. Accessed 14 July 2022.

[98] F. Farazi, J. Akroyd, S. Mosbach, P. Buerger, D. Nurkowski, M. Salamanca, and M. Kraft. OntoKin: An Ontology for Chemical Kinetic Reaction Mechanisms. *J. Chem. Inf. Model.*, 60(1):108–120, 2020. doi:10.1021/acs.jcim.9b00960.

[99] F. Farazi, N. B. Krdzavac, J. Akroyd, S. Mosbach, A. Menon, D. Nurkowski, and M. Kraft. Linking Reaction Mechanisms and Quantum Chemistry: An Ontological Approach. *Comput. Chem. Eng.*, 137:106813, 2020. doi:10.1016/j.compchemeng.2020.106813.

[100] F. Farazi, M. Salamanca, S. Mosbach, J. Akroyd, A. Eibeck, L. K. Aditya, A. Chadzynski, K. Pan, X. Zhou, S. Zhang, M. Q. Lim, and M. Kraft. Knowledge Graph Approach to Combustion Chemistry and Interoperability. *ACS Omega*, 5(29):18342–18348, 2020. doi:10.1021/acsomega.0c02055.

[101] FastAPI Developers. FastAPI Framework, 2023. URL https://github.com/tiangolo/fastapi. Accessed 9 October 2023.

[102] K. C. Felton, J. G. Rittig, and A. A. Lapkin. Summit: Benchmarking Machine Learning Methods for Reaction Optimisation. *Chemistry-Methods*, 1(2):116–122, 2021. doi:10.1002/cmtd.202000051.

[103] T. Finin, R. Fritzson, D. McKay, and R. McEntire. KQML as an Agent Communication Language. In *Proceedings of the Third International Conference on Information and Knowledge Management*, pages 456–463, 1994. doi:10.1145/191246.191322.

[104] D. E. Fitzpatrick and S. V. Ley. Engineering Chemistry for the Future of Chemical Synthesis. *Tetrahedron*, 74(25):3087–3100, 2018. doi:10.1016/j.tet.2017.08.050.

[105] D. E. Fitzpatrick, C. Battilocchio, and S. V. Ley. A Novel Internet-Based Reaction Monitoring, Control and Autonomous Self-Optimization Platform for Chemical Synthesis. *Org. Process Res. Dev.*, 20(2):386–394, 2016. doi:10.1021/acs.oprd.5b00313.

[106] D. E. Fitzpatrick, T. Maujean, A. C. Evans, and S. V. Ley. Across-the-World Automated Optimization and Continuous-Flow Synthesis of Pharmaceutical Agents Operating through a Cloud-Based Server. *Angew. Chem., Int. Ed.*, 57(46):15128–15132, 2018. doi:10.1002/anie.201809080.

[107] D. E. Fitzpatrick, M. O'Brien, and S. V. Ley. A Tutored Discourse on Microcontrollers, Single Board Computers and Their Applications to Monitor and Control Chemical Reactions. *React. Chem. Eng.*, 5(2):201–220, 2020. doi:10.1039/C9RE00407F.

[108] M. M. Flores-Leonar, L. M. Mejía-Mendoza, A. Aguilar-Granda, B. Sanchez-Lengeling, H. Tribukait, C. Amador-Bedolla, and A. Aspuru-Guzik. Materials Acceleration Platforms: On the Way to Autonomous Experimentation. *Curr. Opin. Green Sustain. Chem.*, 25:100370, 2020. doi:10.1016/j.cogsc.2020.100370.

[109] M. Frenklach. Modeling. In W. C. Gardiner, editor, *Combustion Chemistry*, chapter 7, pages 423–453. Springer Verlag, New York, 1984.

[110] M. Frenklach. Transforming Data into Knowledge–Process Informatics for Combustion Chemistry. *Proc. Combust. Inst.*, 31(1):125–140, 2007. doi:10.1016/j.proci.2006.08.121.

[111] M. Frenklach, H. Wang, and M. J. Rabinowitz. Optimization and Analysis of Large Chemical Kinetic Mechanisms Using the Solution Mapping Method - Combustion of Methane. *Prog. Energy Combust. Sci.*, 18(1):47–73, 1992. doi:10.1016/0360-1285(92)90032-V.

[112] J. G. Frey. The Value of the Semantic Web in the Laboratory. *Drug Discov. Today*, 14 (11-12):552–561, 2009. doi:10.1016/j.drudis.2009.03.007.

[113] G. Fu, C. Batchelor, M. Dumontier, J. Hastings, E. Willighagen, and E. Bolton. PubChemRDF: Towards the Semantic Annotation of PubChem Compound and Substance Databases. *J. Cheminf.*, 7:34, 2015. doi:10.1186/s13321-015-0084-4.

[114] J. Galgonek and J. Vondrášek. IDSM ChemWebRDF: SPARQLing Small-Molecule Datasets. *J. Cheminf.*, 13:38, 2021. doi:10.1186/s13321-021-00515-1.

[115] C. W. Gao, J. W. Allen, W. H. Green, and R. H. West. Reaction Mechanism Generator: Automatic Construction of Chemical Kinetic Mechanisms. *Comput. Phys. Commun.*, 203:212–225, 2016. doi:10.1016/j.cpc.2016.02.013.

[116] H. Gao, T. J. Struble, C. W. Coley, Y. Wang, W. H. Green, and K. F. Jensen. Using Machine Learning to Predict Suitable Conditions for Organic Reactions. *ACS Cent. Sci.*, 4(11):1465–1476, 2018. doi:10.1021/acscentsci.8b00357.

[117] D. Garay-Ruiz and C. Bo. Chemical Reaction Network Knowledge Graphs: The OntoRXN Ontology. *J. Cheminf.*, 14(1):29, 2022. doi:10.1186/s13321-022-00610-x.

[118] D. Garijo and Y. Gil. Augmenting PROV with Plans in P-PLAN: Scientific Processes as Linked Data. In *Proceedings of the 2nd International Workshop on Linked Science*, volume 951. CEUR Workshop Proceedings, 2012. URL https://oa.upm.es/19478/.

[119] D. Garijo, Y. Gil, and O. Corcho. Abstract, Link, Publish, Exploit: An End to End Framework for Workflow Sharing. *Future Gener. Comput. Syst.*, 75:271–283, 2017. doi:10.1016/j.future.2017.01.008.

[120] S. S. Garud, I. A. Karimi, and M. Kraft. Design of Computer Experiments: A Review. *Comput. Chem. Eng.*, 106:71–95, 2017. doi:10.1016/j.compchemeng.2017.05.010.

[121] K. Gilmore, D. Kopetzki, J. W. Lee, Z. Horváth, D. T. McQuade, A. Seidel-Morgenstern, and P. H. Seeberger. Continuous Synthesis of Artemisinin-Derived Medicines. *Chem. Commun.*, 50(84):12652–12655, 2014. doi:10.1039/C4CC05098C.

[122] A. Giusti and E. Mastorakos. Turbulent Combustion Modelling and Experiments: Recent Trends and Developments. *Flow, Turbul. Combust.*, 103(4):847–869, 2019. doi:10.1007/s10494-019-00072-6.

[123] G. V. Gkoutos, P. Murray-Rust, H. S. Rzepa, and M. Wright. Chemical Markup, XML, and the World-Wide Web. 3. Toward a Signed Semantic Chemical Web of Trust. *J. Chem. Inf. Comput. Sci.*, 41(5):1124–1130, 2001. doi:10.1021/ci000406v.

[124] B. Glimm, I. Horrocks, B. Motik, G. Stoilos, and Z. Wang. HermiT: An OWL 2 Reasoner. *J. Autom. Reasoning*, 53(3):245–269, 2014. doi:10.1007/s10817-014-9305-1.

[125] C. Goble, S. Cohen-Boulakia, S. Soiland-Reyes, D. Garijo, Y. Gil, M. R. Crusoe, K. Peters, and D. Schober. FAIR Computational Workflows. *Data Intell.*, 2(1-2): 108–121, 2020. doi:10.1162/dint_a_00033.

[126] A. G. Godfrey, T. Masquelin, and H. Hemmerle. A Remote-Controlled Adaptive Medchem Lab: An Innovative Approach to Enable Drug Discovery in the 21st Century. *Drug Discov. Today*, 18(17-18):795–802, 2013. doi:10.1016/j.drudis.2013.03.001.

[127] A. G. Godfrey, S. G. Michael, G. S. Sittampalam, and G. Zahoránszky-Köhalmi. A Perspective on Innovating the Chemistry Lab Bench. *Front. Robot. AI*, 7:24, 2020. doi:10.3389/frobt.2020.00024.

[128] C. Gomes, T. Dietterich, C. Barrett, J. Conrad, B. Dilkina, S. Ermon, F. Fang, A. Farnsworth, A. Fern, X. Fern, D. Fink, D. Fisher, A. Flecker, D. Freund, A. Fuller, J. Gregoire, J. Hopcroft, S. Kelling, Z. Kolter, W. Powell, N. Sintov, J. Selker, B. Selman, D. Sheldon, D. Shmoys, M. Tambe, W.-K. Wong, C. Wood, X. Wu, Y. Xue, A. Yadav, A.-A. Yakubu, and M. L. Zeeman. Computational Sustainability: Computing for a Better World and a Sustainable Future. *Commun. ACM*, 62(9):56–65, 2019. doi:10.1145/3339399.

[129] C. P. Gomes, J. Bai, Y. Xue, J. Björck, B. Rappazzo, S. Ament, R. Bernstein, S. Kong, S. K. Suram, R. B. van Dover, and J. M. Gregoire. CRYSTAL: A Multi-Agent AI System for Automated Mapping of Materials' Crystal Structures. *MRS Commun.*, 9 (2):600–608, 2019. doi:10.1557/mrc.2019.50.

[130] J. Goodman. Computer Software Review: Reaxys. *J. Chem. Inf. Model.*, 49(12): 2897–2898, 2009. doi:10.1021/ci900437n.

[131] D. G. Goodwin, R. L. Speth, H. K. Moffat, and B. W. Weber. Cantera: An Object-oriented Software Toolkit for Chemical Kinetics, Thermodynamics, and Transport Processes. https://www.cantera.org, 2018. Version 2.4.0. Accessed 6 March 2021.

[132] Google. Google Knowledge Graph Search API, 2022. URL https://developers.google.com/knowledge-graph. Accessed 29 September 2023.

[133] R. L. Greenaway, V. Santolini, M. J. Bennison, B. M. Alston, C. J. Pugh, M. A. Little, M. Miklitz, E. G. B. Eden-Rump, R. Clowes, A. Shakil, H. J. Cuthbertson, H. Armstrong, M. E. Briggs, K. E. Jelfs, and A. I. Cooper. High-Throughput Discovery of Organic Cages and Catenanes Using Computational Screening Fused with Robotic Synthesis. *Nat. Commun.*, 9(1):2849, 2018. doi:10.1038/s41467-018-05271-9.

[134] G. Grethe, G. Blanke, H. Kraut, and J. M. Goodman. International Chemical Identifier for Reactions (RInChI). *J. Cheminf.*, 10(1):22, 2018. doi:10.1186/s13321-018-0277-8.

[135] T. R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowl. Acquis.*, 5(2):199–220, 1993. doi:10.1006/knac.1993.1008.

[136] D. Guevarra, K. Kan, Y. Lai, R. Jones, L. Zhou, P. Donnelly, M. Richter, H. Stein, and J. Gregoire. Orchestrating Nimble Experiments Across Interconnected Labs. *ChemRxiv Preprint*, 2023. doi:10.26434/chemrxiv-2023-jgc8g.

[137] R. V. Guha, D. Brickley, and S. Macbeth. Schema.org: Evolution of Structured Data on the Web. *Commun. ACM*, 59(2):44–51, 2016. doi:10.1145/2844544.

[138] J. Guo, A. S. Ibanez-Lopez, H. Gao, V. Quach, C. W. Coley, K. F. Jensen, and R. Barzilay. Automated Chemical Reaction Extraction from Scientific Literature. *J. Chem. Inf. Model.*, 62(9):2035–2045, 2021. doi:10.1021/acs.jcim.1c00284.

[139] C. Gutierrez and J. F. Sequeda. Knowledge Graphs. *Commun. ACM*, 64(3):96–104, 2021. doi:10.1145/3418294.

[140] J. Hachmann, M. A. F. Afzal, M. Haghighatlari, and Y. Pal. Building and Deploying a Cyberinfrastructure for the Data-driven Design of Chemical Systems and the Exploration of Chemical Space. *Mol. Simul.*, 44(11):921–929, 2018. doi:10.1080/08927022.2018.1471692.

[141] M. Haghighatlari, G. Vishwakarma, D. Altarawy, R. Subramanian, B. U. Kota, A. Sonpal, S. Setlur, and J. Hachmann. ChemML: A Machine Learning and Informatics Program Package for the Analysis, Mining, and Modeling of Chemical and Materials Data. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.*, 10(4):e1458, 2020. doi:10.1002/wcms.1458.

[142] M. HamediRad, R. Chao, S. Weisberg, J. Lian, S. Sinha, and H. Zhao. Towards a Fully Automated Algorithm Driven Platform for Biosystems Design. *Nat. Commun.*, 10(1):1–10, 2019. doi:10.1038/s41467-019-13189-z.

[143] A. J. S. Hammer, A. I. Leonov, N. L. Bell, and L. Cronin. Chemputation and the Standardization of Chemical Informatics. *JACS Au*, 1(10):1572–1587, 2021. doi:10.1021/jacsau.1c00303.

[144] M. Hammond, A. Chen, S. Djordjević, D. Butler, and O. Mark. Urban Flood Impact Assessment: A State-of-the-Art Review. *Urban Water J.*, 12(1):14–29, 2013. doi:10.1080/1573062x.2013.857421.

[145] O. Hartig and B. Thompson. Foundations of an Alternative Approach to Reification in RDF. *arXiv Preprint*, 2021. doi:10.48550/arXiv.1406.3399.

[146] F. Häse, L. M. Roch, and A. Aspuru-Guzik. Next-Generation Experimentation with Self-Driving Laboratories. *Trends Chem.*, 1(3):282–291, 2019. doi:10.1016/j.trechm.2019.02.007.

[147] J. Hastings, L. Chepelev, E. Willighagen, N. Adams, C. Steinbeck, and M. Dumontier. The Chemical Information Ontology: Provenance and Disambiguation for Chemical Data on the Biological Semantic Web. *PLoS One*, 6(10):e25513, 2011. doi:10.1371/journal.pone.0025513.

[148] J. Hastings, P. de Matos, A. Dekker, M. Ennis, B. Harsha, N. Kale, V. Muthukrishnan, G. Owen, S. Turner, M. Williams, and C. Steinbeck. The ChEBI Reference Database and Ontology for Biologically Relevant Chemistry: Enhancements for 2013. *Nucleic Acids Res.*, 41(D1):D456–D463, 2012. doi:10.1093/nar/gks1146.

[149] J. Hastings, M. Glauer, A. Memariani, F. Neuhaus, and T. Mossakowski. Learning Chemistry: Exploring the Suitability of Machine Learning for the Task of Structure-Based Chemical Ontology Classification. *J. Cheminf.*, 13:23, 2021. doi:10.1186/s13321-021-00500-8.

[150] T. He, H. Liu, Y. Wang, B. Wang, H. Liu, and Z. Wang. Development of Surrogate Model for Oxygenated Wide-distillation Fuel with Polyoxymethylene Dimethyl Ether. *SAE Int. J. Fuels Lubr.*, 10(3):803–814, 2017. doi:10.4271/2017-01-2336.

[151] T. He, Z. Wang, X. You, H. Liu, Y. Wang, X. Li, and X. He. A Chemical Kinetic Mechanism for the Low- and Intermediate-Temperature Combustion of Polyoxymethylene Dimethyl Ether 3 (PODE3). *Fuel*, 212:223–235, 2018. doi:10.1016/j.fuel.2017.09.080.

[152] J. Hein, R. Rauschen, M. Guy, and L. Cronin. Universal Chemical Programming Language for Robotic Synthesis Reproducibility, 2023. Research Square Platform LLC. doi:10.21203/rs.3.rs-2761997/v1.

[153] S. R. Heller, A. McNaught, I. Pletnev, S. Stein, and D. Tchekhovskoi. InChI, the IUPAC International Chemical Identifier. *J. Cheminf.*, 7:23, 2015. doi:10.1186/s13321-015-0068-4.

[154] J. Hendler. Agents and the Semantic Web. *IEEE Intell. Syst.*, 16(2):30–37, 2001. doi:10.1109/5254.920597.

[155] J. Hendler. Semantic Web: The Inside Story. Summer School in Cognitive Science: Web Science and the Mind, Institut des Sciences Cognitives, Université du Québec à Montréal, Montréal, Canada, 2014. URL https://www.slideshare.net/jahendler/semantic-web-the-inside-story. A recording is available at https://www.youtube.com/watch?v=3Ap5FsxvjTQ. Accessed 18 September 2023.

[156] S. Herres-Pawlis, O. Koepler, and C. Steinbeck. NFDI4Chem: Shaping a Digital and Cultural Change in Chemistry. *Angew. Chem., Int. Ed.*, 58(32):10766–10768, 2019. doi:10.1002/anie.201907260.

[157] P. Hitzler. A Review of The Semantic Web Field. *Commun. ACM*, 64(2):76–83, 2021. doi:10.1145/3397512.

[158] HM Land Registry. HM Land Registry Open Data, 2022. URL https://landregistry.data.gov.uk/. Accessed 13 Oct 2022.

[159] M. Hofmeister, S. Mosbach, J. Hammacher, M. Blum, G. Röhrig, C. Dörr, V. Flegel, A. Bhave, and M. Kraft. Resource-Optimised Generation Dispatch Strategy for District Heating Systems Using Dynamic Hierarchical Optimisation. *Appl. Energy*, 305:117877, 2022. doi:10.1016/j.apenergy.2021.117877.

[160] M. Hofmeister, J. Bai, G. Brownbridge, S. Mosbach, K. F. Lee, F. Farazi, M. Hillman, M. Agarwal, S. Ganguly, J. Akroyd, and M. Kraft. Semantic Agent Framework for Automated Flood Assessment Using Dynamic Knowledge Graphs, 2023. Submitted for publication. Preprint available from https://como.ceb.cam.ac.uk/preprints/309/.

[161] A. Hogan, E. Blomqvist, M. Cochez, C. D'Amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A.-C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann. Knowledge Graphs. *ACM Comput. Surv.*, 54(4):1–37, 2022. doi:10.1145/3447772.

[162] I. Holland and J. A. Davies. Automation in the Life Science Research Laboratory. *Front. Bioeng. Biotechnol.*, 8:571777, 2020. doi:10.3389/fbioe.2020.571777.

[163] R. Hoogenboom, M. W. M. Fijten, C. Brändli, J. Schroer, and U. S. Schubert. Automated Parallel Temperature Optimization and Determination of Activation Energy for the Living Cationic Polymerization of 2-Ethyl-2-Oxazoline. *Macromol. Rapid Commun.*, 24(1):98–103, 2003. doi:10.1002/marc.200390017.

[164] R. Hooke and T. A. Jeeves. "Direct Search" Solution of Numerical and Statistical Problems. *J. ACM*, 8(2):212–229, 1961. doi:10.1145/321062.321069.

[165] M. Horridge and S. Bechhofer. The OWL API: A Java API for OWL Ontologies. *Semant. Web*, 2(1):11–21, 2011. doi:10.3233/sw-2011-0025.

[166] M. Horridge and P. F. Patel-Schneider. OWL 2 Web Ontology Language Manchester Syntax (Second Edition). W3C Working Group Note 11 December 2012, 2012. URL https://www.w3.org/TR/2012/NOTE-owl2-manchester-syntax-20121211/. Accessed 15 September 2023.

[167] B. Huang, G. F. von Rudorff, and O. A. von Lilienfeld. The Central Role of Density Functional Theory in the AI Age. *Science*, 381(6654):170–175, 2023. doi:10.1126/science.abn3445.

[168] S. P. Huber, S. Zoupanos, M. Uhrin, L. Talirz, L. Kahle, R. Häuselmann, D. Gresch, T. Müller, A. V. Yakutovich, C. W. Andersen, F. F. Ramirez, C. S. Adorf, F. Gargiulo, S. Kumbhar, E. Passaro, C. Johnston, A. Merkys, A. Cepellotti, N. Mounet, N. Marzari, B. Kozinsky, and G. Pizzi. AiiDA 1.0, a Scalable Computational Infrastructure for Automated Reproducible Workflows and Data Provenance. *Sci. Data*, 7(1):300, 2020. doi:10.1038/s41597-020-00638-4.

[169] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Parallel Algorithm Configuration. In *International Conference on Learning and Intelligent Optimization*, pages 55–70. Springer, 2012. doi:10.1007/978-3-642-34413-8_5.

[170] O. Inderwildi, C. Zhang, X. Wang, and M. Kraft. The Impact of Intelligent Cyber-Physical Systems on the Decarbonization of Energy. *Energy Environ. Sci.*, 13(3): 744–771, 2020. doi:10.1039/C9EE01919G.

[171] R. J. Ingham, C. Battilocchio, J. M. Hawkins, and S. V. Ley. Integration of Enabling Methods for the Automated Flow Preparation of Piperazine-2-Carboxamide. *Beilstein J. Org. Chem.*, 10(1):641–652, 2014. doi:10.3762/bjoc.10.56.

[172] R. J. Ingham, C. Battilocchio, D. E. Fitzpatrick, E. Sliwinski, J. M. Hawkins, and S. V. Ley. A Systems Approach Towards an Intelligent and Self-Controlling Platform for Integrated Continuous Reaction Sequences. *Angew. Chem., Int. Ed.*, 127(1):146–150, 2015. doi:10.1002/anie.201409356.

[173] Insight Centre for Data Analytics. The Linked Open Data Cloud, 2023. URL https://lod-cloud.net/. Accessed 15 September 2023.

[174] K. M. Jablonka, L. Patiny, and B. Smit. Making the Collective Knowledge of Chemistry Open and Machine Actionable. *Nat. Chem.*, 14(4):365–376, 2022. doi:10.1038/s41557-022-00910-7.

[175] S. Jacobs, M. Döntgen, A. B. S. Alquaity, W. A. Kopp, L. C. Kröger, U. Burke, H. Pitsch, K. Leonhard, H. J. Curran, and K. A. Heufer. Detailed Kinetic Modeling of Dimethoxymethane. Part II: Experimental and Theoretical Study of the Kinetics and Reaction Mechanism. *Combust. Flame*, 205:522–533, 2019. doi:10.1016/j.combustflame.2018.12.026.

[176] JADE. Java Agent DEvelopment Framework: Jade Site, 2022. URL https://jade.tilab.com/. Accessed 14 July 2022.

[177] A. Jain, S. P. Ong, W. Chen, B. Medasani, X. Qu, M. Kocher, M. Brafman, G. Petretto, G.-M. Rignanese, G. Hautier, D. Gunter, and K. A. Persson. FireWorks: A Dynamic Workflow System Designed for High-Throughput Applications. *Concurr. Comput. Pract. Exp*, 27(17):5037–5059, 2015. doi:10.1002/cpe.3505.

[178] M. I. Jeraal, S. Sung, and A. A. Lapkin. A Machine Learning-Enabled Autonomous Flow Chemistry Platform for Process Optimization of Multiple Reaction Metrics. *Chemistry-Methods*, 1(1):71–77, 2021. doi:10.1002/cmtd.202000044.

[179] W. Jin, C. W. Coley, R. Barzilay, and T. Jaakkola. Predicting Organic Reaction Outcomes with Weisfeiler-Lehman Network. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 2604–2613, 2017. URL https://dl.acm.org/doi/abs/10.5555/3294996.3295021.

[180] N. Jung. Documentation and Publication of Reactions with Chemotion ELN and Repository. NIH Virtual Workshop on Reaction Informatics, May 18-20, 2021. URL https://cactus.nci.nih.gov/presentations/NIHReactInf_2021-05/Nicole_Jung_Chemotion_NIH_2021.pdf. Accessed 13 November 2021.

[181] C. A. Kastner, A. Braumann, P. L. W. Man, S. Mosbach, G. P. E. Brownbridge, J. Akroyd, M. Kraft, and C. Himawan. Bayesian Parameter Estimation for a Jet-milling Model Using Metropolis-Hastings and Wang-Landau Sampling. *Chem. Eng. Sci.*, 89:244–257, 2013. doi:10.1016/j.ces.2012.11.027.

[182] R. F. Kazmierczak Jr. Optimizing Complex Bioeconomic Simulations Using an Efficient Search Heuristic. *DAE Research Report No. 704C61*, pages 1–38, 1996. doi:10.2139/ssrn.15071.

[183] S. Kearnes. The Open Reaction Database. NIH Virtual Workshop on Reaction Informatics, May 18-20, 2021. URL https://cactus.nci.nih.gov/presentations/NIHReactInf_2021-05/Kearnes_Open_Reaction_Database-NIH_Reaction_Informatics.pptx. Accessed 13 November 2021.

[184] S. M. Kearnes, M. R. Maser, M. Wleklinski, A. Kast, A. G. Doyle, S. D. Dreher, J. M. Hawkins, K. F. Jensen, and C. W. Coley. The Open Reaction Database. *J. Am. Chem. Soc.*, 143(45):18820–18826, 2021. doi:10.1021/jacs.1c09820.

[185] R. J. Kee, F. M. Rupley, , E. Meeks, and J. A. Miller. CHEMKIN-III: A FORTRAN Chemical Kinetics Package for the Analysis of Gas-Phase Chemical and Plasma Kinetics. Sandia Natl. Lab. [Tech. Rep.] SAND 96-8216, Sandia National Laboratories, 1996.

[186] M. Kifer and H. Boley. RIF Overview (Second Edition). W3C Working Group Note 5 February 2013, 2013. URL https://www.w3.org/TR/rif-overview/. Accessed 15 September 2023.

[187] S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang, and E. E. Bolton. PubChem 2019 Update: Improved Access to Chemical Data. *Nucleic Acids Res.*, 47(D1):D1102–D1109, 2019. doi:10.1093/nar/gky1033.

[188] R. King, Y. Gil, and H. Kitano. The Turing AI Scientist Grand Challenge, 2023. URL https://www.turing.ac.uk/research/research-projects/turing-ai-scientist-grand-challenge. Accessed 27 September 2023.

[189] R. D. King, K. E. Whelan, F. M. Jones, P. G. K. Reiser, C. H. Bryant, S. H. Muggleton, D. B. Kell, and S. G. Oliver. Functional Genomic Hypothesis Generation and Experimentation by a Robot Scientist. *Nature*, 427(6971):247–252, 2004. doi:10.1038/nature02236.

[190] R. D. King, J. Rowland, S. G. Oliver, M. Young, W. Aubrey, E. Byrne, M. Liakata, M. Markham, P. Pir, L. N. Soldatova, A. Sparkes, K. E. Whelan, and A. Clare. The Automation of Science. *Science*, 324(5923):85–89, 2009. doi:10.1126/science.1165620.

[191] H. Kitano. Nobel Turing Challenge: Creating the Engine for Scientific Discovery. *npj Syst. Biol. Appl.*, 7(1):1–12, 2021. doi:10.1038/s41540-021-00189-3.

[192] N. J. Knight, S. Kanza, D. Cruickshank, W. S. Brocklesby, and J. G. Frey. Talk2Lab: The Smart Lab of the Future. *IEEE Internet Things J.*, 7(9):8631–8640, 2020. doi:10.1109/JIOT.2020.2995323.

[193] K. Kohse-Höinghaus, M. Reimann, and J. Guzy. Clean Combustion: Chemistry and Diagnostics for a Systems Approach in Transportation and Energy Conversion. *Prog. Energy Combust. Sci.*, 65(1), 2018. doi:10.1016/j.pecs.2017.10.001.

[194] A. Kondinski, A. Menon, D. Nurkowski, F. Farazi, S. Mosbach, J. Akroyd, and M. Kraft. Automated Rational Design of Metal–Organic Polyhedra. *J. Am. Chem. Soc.*, 144(26):11713–11728, 2022. doi:10.1021/jacs.2c03402.

[195] M. Kraft and S. Mosbach. The Future of Computational Modelling in Reaction Engineering. *Philos. Trans. R. Soc., A*, 368(1924):3633–3644, 2010. doi:10.1098/rsta.2010.0124.

[196] M. Krämer, H. M. Würz, and C. Altenhofen. Executing Cyclic Scientific Workflows in the Cloud. *J. Cloud Comput.*, 10(1):1–26, 2021. doi:10.1186/s13677-021-00229-7.

[197] N. Krdzavac, S. Mosbach, D. Nurkowski, P. Buerger, J. Akroyd, J. Martin, A. Menon, and M. Kraft. An Ontology and Semantic Web Service for Quantum Chemistry Calculations. *J. Chem. Inf. Model.*, 59(7):3154–3165, 2019. doi:10.1021/acs.jcim.9b00227.

[198] D. Krech, G. A. Grimnes, G. Higgins, J. Hees, I. Aucamp, N. Lindström, N. Arndt, A. Sommer, E. Chuc, I. Herman, A. Nelson, J. McCusker, T. Gillespie, T. Kluyver, F. Ludwig, P.-A. Champin, M. Watts, U. Holzer, E. Summers, W. Morriss, D. Winston, D. Perttula, F. Kovacevic, R. Chateauneu, H. Solbrig, B. Cogrel, and V. Stuart. RDFLib, 2023. URL https://github.com/RDFLib/rdflib. Accessed 29 September 2023.

[199] M. Krenn, F. Häse, A. Nigam, P. Friederich, and A. Aspuru-Guzik. Self-Referencing Embedded Strings (SELFIES): A 100% Robust Molecular String Representation. *Mach. Learn.: Sci. Technol.*, 1(4):045024, 2020. doi:10.1088/2632-2153/aba947.

[200] A. G. Kusne, H. Yu, C. Wu, H. Zhang, J. Hattrick-Simpers, B. DeCost, S. Sarker, C. Oses, C. Toher, S. Curtarolo, A. V. Davydov, R. Agarwal, L. A. Bendersky, M. Li, A. Mehta, and I. Takeuchi. On-the-Fly Closed-Loop Materials Discovery via Bayesian Active Learning. *Nat. Commun.*, 11(1):1–11, 2020. doi:10.1038/s41467-020-19597-w.

[201] P. Lampen, J. Lambert, R. J. Lancashire, R. S. McDonald, P. S. McIntyre, D. N. Rutledge, T. Fröhlich, and A. N. Davies. An Extension to the JCAMP-DX Standard File Format, JCAMP-DX V. 5.01. *Pure Appl. Chem.*, 71(8):1549–1556, 1999. doi:10.1351/pac199971081549.

[202] G. Landrum. RDKit: Open-Source Cheminformatics Software, 2023. URL https://www.rdkit.org/. Accessed 17 July 2023.

[203] S. Langner, F. Häse, J. D. Perea, T. Stubhan, J. Hauch, L. M. Roch, T. Heumueller, A. Aspuru-Guzik, and C. J. Brabec. Beyond Ternary OPV: High-Throughput Experimentation and Self-Driving Laboratories Optimize Multicomponent Systems. *Adv. Mater.*, 32(14):1907801, 2020. doi:10.1002/adma.201907801.

[204] T. Lebo, S. Sahoo, D. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, and J. Zhao. PROV-O: The PROV Ontology. W3C Recommendation, 2013. URL https://www.w3.org/TR/prov-o/. Accessed 29 September 2023.

[205] J. Lederberg, G. L. Sutherland, B. G. Buchanan, E. A. Feigenbaum, A. V. Robertson, A. M. Duffield, and C. Djerassi. Applications of Artificial Intelligence for Chemical Inference. I. Number of Possible Organic Compounds. Acyclic Structures Containing Carbon, Hydrogen, Oxygen, and Nitrogen. *J. Am. Chem. Soc.*, 91(11):2973–2976, 1969. doi:10.1021/ja01039a025.

[206] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia – A Large-Scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semant. Web*, 6(2):167–195, 2015. doi:10.3233/SW-140134.

[207] D. A. Leins, S. B. Haase, M. Eslami, J. Schrier, and J. T. Freeman. Collaborative Methods to Enhance Reproducibility and Accelerate Discovery. *Digital Discovery*, 2 (1):12–27, 2023. doi:10.1039/D2DD00061J.

[208] S. V. Ley, D. E. Fitzpatrick, R. J. Ingham, and R. M. Myers. Organic Synthesis: March of the Machines. *Angew. Chem., Int. Ed.*, 54(11):3449–3464, 2015. doi:10.1002/anie.201410744.

[209] J. Li, J. Li, R. Liu, Y. Tu, Y. Li, J. Cheng, T. He, and X. Zhu. Autonomous Discovery of Optically Active Chiral Inorganic Perovskite Nanocrystals through an Intelligent Cloud Lab. *Nat. Commun.*, 11:2046, 2020. doi:10.1038/s41467-020-15728-5.

[210] J. Li, Y. Tu, R. Liu, Y. Lu, and X. Zhu. Toward "On-Demand" Materials Synthesis and Scientific Discovery through Intelligent Robots. *Adv. Sci.*, 7(7):1901957, 2020. doi:10.1002/advs.201901957.

[211] Y.-P. Li, K. Han, C. A. Grambow, and W. H. Green. Self-Evolving Machine: A Continuously Improving Model for Molecular Thermochemistry. *J. Phys. Chem. A*, 123(10):2142–2152, 2019. doi:10.1021/acs.jpca.8b10789.

[212] Z. Li, M. A. Najeeb, L. Alves, A. Z. Sherman, V. Shekar, P. Cruz Parrilla, I. M. Pendleton, W. Wang, P. W. Nega, M. Zeller, J. Schrier, A. J. Norquist, and E. M. Chan. Robot-Accelerated Perovskite Investigation and Discovery. *Chem. Mater.*, 32(13): 5650–5663, 2020. doi:10.1021/acs.chemmater.0c01153.

[213] Q. Lin, K. L. Tay, D. Zhou, and W. Yang. Development of a Compact and Robust Polyoxymethylene Dimethyl Ether 3 Reaction Mechanism for Internal Combustion Engines. *Energy Convers. Manage.*, 185:35–43, 2019. doi:10.1016/j.enconman.2019.02.007.

[214] J. S. Lindsey. A Retrospective on the Automation of Laboratory Synthetic Chemistry. *Chemom. Intell. Lab. Syst.*, 17(1):15–45, 1992. doi:10.1016/0169-7439(92)90025-B.

[215] H. Liu, Z. Wang, J. Wang, and X. He. Improvement of Emission Characteristics and Thermal Efficiency in Diesel Engines by Fueling Gasoline/Diesel/PODEn Blends. *Energy*, 97:105–112, 2016. doi:10.1016/j.energy.2015.12.110.

[216] J. Liu, E. Pacitti, P. Valduriez, and M. Mattoso. A Survey of Data-Intensive Scientific Workflow Management. *J. Grid Comput.*, 13(4):457–493, 2015. doi:10.1007/s10723-015-9329-8.

[217] S. Lo, S. Baird, J. Schrier, B. Blaiszik, S. Kalinin, H. Tran, T. Sparks, and A. Aspuru-Guzik. Review of Low-Cost Self-Driving Laboratories: The "Frugal Twin" Concept. *ChemRxiv Preprint*, 2023. doi:10.26434/chemrxiv-2023-6z9mq.

[218] N. Lopes, A. Zimmermann, A. Hogan, G. Lukácsy, A. Polleres, U. Straccia, and S. Decker. RDF Needs Annotations. W3C Workshop – RDF Next Steps, Stanford, CA, USA, June 26-27, 2010. URL https://www.w3.org/2009/12/rdf-ws/papers/ws09. Accessed 29 September 2023.

[219] D. Lowe. Chemical Reactions from US Patents (1976-Sep2016), 2017. URL https://figshare.com/articles/dataset/Chemical_reactions_from_US_patents_1976-Sep2016_/5104873/1.

[220] E. Lyons, G. Papadimitriou, C. Wang, K. Thareja, P. Ruth, J. Villalobos, I. Rodero, E. Deelman, M. Zink, and A. Mandal. Toward a Dynamic Network-Centric Distributed Cloud Platform for Scientific Workflows: A Case Study for Adaptive Weather Sensing. In *2019 15th International Conference on eScience (eScience)*, pages 67–76. IEEE, 2019. doi:10.1109/eScience.2019.00015.

[221] B. P. MacLeod, F. G. L. Parlane, T. D. Morrissey, F. Häse, L. M. Roch, K. E. Dettelbach, R. Moreira, L. P. E. Yunker, M. B. Rooney, J. R. Deeth, V. Lai, G. J. Ng, H. Situ, R. H. Zhang, M. S. Elliott, T. H. Haley, D. J. Dvorak, A. Aspuru-Guzik, J. E. Hein, and C. P. Berlinguette. Self-Driving Laboratory for Accelerated Discovery of Thin-Film Materials. *Sci. Adv.*, 6(20):eaaz8867, 2020. doi:10.1126/sciadv.aaz8867.

[222] P. M. Maffettone, S. Campbell, M. D. Hanwell, S. Wilkins, and D. Olds. Delivering Real-Time Multi-Modal Materials Analysis with Enterprise Beamlines. *Cell Rep. Phys. Sci.*, 3(11):101112, 2022. doi:10.1016/j.xcrp.2022.101112.

[223] P. M. Maffettone, P. Friederich, S. G. Baird, B. Blaiszik, K. A. Brown, S. I. Campbell, O. A. Cohen, T. Collins, R. L. Davis, I. T. Foster, N. Haghmoradi, M. Hereld, N. Jung, H.-K. Kwon, G. Pizzuto, J. Rintamaki, C. Steinmann, L. Torresi, and S. Sun. What is Missing in Autonomous Discovery: Open Challenges for the Community. *arXiv Preprint*, 2023. doi:10.48550/arXiv.2304.11120.

[224] D. Marquez-Gamez and P. Maffetton. A ROS Based Architecture for an Autonomous Chemistry Laboratory. In *ROSCon Macau 2019*. Open Robotics, October 2019. doi:10.36288/ROSCon2019-900915. URL https://doi.org/10.36288/ROSCon2019-900915.

[225] C. Mateos, M. J. Nieves-Remacha, and J. A. Rincón. Automated Platforms for Reaction Self-Optimization in Flow. *React. Chem. Eng.*, 4(9):1536–1544, 2019. doi:10.1039/C9RE00116F.

[226] K. Mathew, J. H. Montoya, A. Faghaninia, S. Dwarakanath, M. Aykol, H. Tang, I.-h. Chu, T. Smidt, B. Bocklund, M. Horton, J. Dagdelen, B. Wood, Z. K. Liu, J. Neaton, S. P. Ong, K. Persson, and A. Jain. Atomate: A High-Level Interface to Generate, Execute, and Analyze Computational Materials Science Workflows. *Comput. Mater. Sci.*, 139:140–152, 2017. doi:10.1016/j.commatsci.2017.07.030.

[227] A. McNally, C. K. Prier, and D. W. MacMillan. Discovery of an α-Amino C–H Arylation Reaction Using the Strategy of Accelerated Serendipity. *Science*, 334(6059): 1114–1117, 2011. doi:10.1126/science.1213920.

[228] A. McNally, B. Haffemayer, B. S. L. Collins, and M. J. Gaunt. Palladium-Catalysed C–H Activation of Aliphatic Amines to Give Strained Nitrogen Heterocycles. *Nature*, 510(7503):129–133, 2014. doi:10.1038/nature13389.

[229] S. H. M. Mehr, M. Craven, A. I. Leonov, G. Keenan, and L. Cronin. A Universal System for Digitization and Automatic Execution of the Chemical Synthesis Literature. *Science*, 370(6512):101–108, 2020. doi:10.1126/science.abc2986.

[230] A. Menon, N. B. Krdzavac, and M. Kraft. From Database to Knowledge Graph—Using Data in Chemistry. *Curr. Opin. Chem. Eng.*, 26:33–37, 2019. doi:10.1016/j.coche.2019.08.004.

[231] R. Mercado, S. M. Kearnes, and C. W. Coley. Data Sharing in Chemistry: Lessons Learned and a Case for Mandating Structured Reaction Data. *J. Chem. Inf. Model.*, 2023. doi:10.1021/acs.jcim.3c00607.

[232] R. B. Merrifield. Automated Synthesis of Peptides. *Science*, 150(3693):178–185, 1965. doi:10.1126/science.150.3693.178.

[233] R. B. Merrifield, J. M. Stewart, and N. Jernberg. Instrument for Automated Synthesis of Peptides. *Anal. Chem.*, 38(13):1905–1914, 1966. doi:10.1021/ac50155a057.

[234] B. Miles and P. L. Lee. Achieving Reproducibility and Closed-Loop Automation in Biological Experimentation with an IoT-Enabled Lab of the Future. *SLAS Technol.*, 23(5):432–439, 2018. doi:10.1177/2472630318784506.

[235] T. Millecam, A. J. Jarrett, N. Young, D. E. Vanderwall, and D. Della Corte. Coming of Age of Allotrope: Proceedings from the Fall 2020 Allotrope Connect. *Drug Discov. Today*, 26(8):1922–1928, 2021. doi:10.1016/j.drudis.2021.03.028.

[236] M. Minsky. *The Society of Mind*. Simon & Schuster, Inc., 1988.

[237] S. N. Mitchell, A. Lahiff, N. Cummings, J. Hollocombe, B. Boskamp, R. Field, D. Reddyhoff, K. Zarebski, A. Wilson, B. Viola, M. Burke, B. Archibald, P. Bessell, R. Blackwell, L. A. Boden, A. Brett, S. Brett, R. Dundas, J. Enright, A. N. Gonzalez-Beltran, C. Harris, I. Hinder, C. D. Hughes, M. Knight, V. Mano, C. McMonagle, D. Mellor, S. Mohr, G. Marion, L. Matthews, I. J. McKendrick, C. M. Pooley, T. Porphyre, A. Reeves, E. Townsend, R. Turner, J. Walton, and R. Reeve. FAIR Data Pipeline: Provenance-Driven Data Management for Traceable Scientific Workflows. *Philos. Trans. R. Soc., A*, 380(2233):20210300, 2022. doi:10.1098/rsta.2021.0300.

[238] Y. Mo, G. Rughoobur, A. M. K. Nambiar, K. Zhang, and K. F. Jensen. A Multifunctional Microfluidic Platform for High-Throughput Experimentation of Electroorganic Chemistry. *Angew. Chem., Int. Ed.*, 59(47):20890–20894, 2020. doi:10.1002/anie.202009819.

[239] N. Mohammadi and J. E. Taylor. Thinking Fast and Slow in Disaster Decision-Making with Smart City Digital Twins. *Nat. Comput. Sci.*, 1(12):771–773, 2021. doi:10.1038/s43588-021-00174-0.

[240] MolSSI QCArchive. The MolSSI Quantum Chemistry Archive, 2023. URL https://qcarchive.molssi.org/. Accessed 17 July 2023.

[241] J. H. Montoya, K. T. Winther, R. A. Flores, T. Bligaard, J. S. Hummelshøj, and M. Aykol. Autonomous Intelligent Agents for Accelerated Materials Discovery. *Chem. Sci.*, 11(32):8517–8532, 2020. doi:10.1039/D0SC01101K.

[242] J. Morbach, A. Yang, and W. Marquardt. OntoCAPE - A Large-Scale Ontology for Chemical Process Engineering. *Eng. Appl. Artif. Intell.*, 20(2):147–161, 2007. doi:10.1016/j.engappai.2006.06.010.

[243] L. Moreau, B. Ludäscher, I. Altintas, R. S. Barga, S. Bowers, S. Callahan, G. Chin, B. Clifford, S. Cohen, S. Cohen-Boulakia, S. Davidson, E. Deelman, L. Digiampietri, I. Foster, J. Freire, J. Frew, J. Futrelle, T. Gibson, Y. Gil, C. Goble, J. Golbeck, P. Groth, D. A. Holland, S. Jiang, J. Kim, D. Koop, A. Krenek, T. McPhillips, G. Mehta, S. Miles, D. Metzger, S. Munroe, J. Myers, B. Plale, N. Podhorszki, V. Ratnakar, E. Santos, C. Scheidegger, K. Schuchardt, M. Seltzer, Y. L. Simmhan, C. Silva, P. Slaughter,

E. Stephan, R. Stevens, D. Turi, H. Vo, M. Wilde, J. Zhao, and Y. Zhao. Special Issue: The First Provenance Challenge. *Concurr. Comput.-Pract. Exp.*, 20(5):409–418, 2008. doi:10.1002/cpe.1233.

[244] L. Moreau, L. Ding, J. Futrelle, D. G. Verdejo, P. Groth, M. Jewell, S. Miles, P. Missier, J. Pan, and J. Zhao. Open Provenance Model (OPM) OWL Specification, 2010. URL https://openprovenance.org/opm/model/opmo. Accessed 29 September 2023.

[245] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, B. Plale, Y. Simmhan, E. Stephan, and J. V. den Bussche. The Open Provenance Model Core Specification (v1.1). *Future Gener. Comput. Syst.*, 27(6):743–756, 2011. doi:10.1016/j.future.2010.07.005.

[246] S. Mosbach, A. Braumann, P. L. W. Man, C. A. Kastner, G. P. E. Brownbridge, and M. Kraft. Iterative Improvement of Bayesian Parameter Estimates for an Engine Model by Means of Experimental Design. *Combust. Flame*, 159(3):1303–1313, 2012. doi:10.1016/j.combustflame.2011.10.019.

[247] S. Mosbach, J. H. Hong, G. P. E. Brownbridge, M. Kraft, S. Gudiyella, and K. Brezinsky. Bayesian Error Propagation for a Kinetic Model of n-Propylbenzene Oxidation in a Shock Tube. *Int. J. Chem. Kinet.*, 46(7):389–404, 2014. doi:10.1002/kin.20855.

[248] S. Mosbach, A. Menon, F. Farazi, N. Krdzavac, X. Zhou, J. Akroyd, and M. Kraft. Multiscale Cross-Domain Thermochemical Knowledge-Graph. *J. Chem. Inf. Model.*, 60(12):6155–6166, 2020. doi:10.1021/acs.jcim.0c01145.

[249] B. Motik, B. Parsia, P. F. Patel-Schneider, S. Bechhofer, B. C. Grau, A. Fokoue, and R. Hoekstra. OWL 2 Web Ontology Language XML Serialization (Second Edition). W3C Recommendation 11 December 2012, 2012. URL https://www.w3.org/TR/2012/REC-owl2-xml-serialization-20121211/. Accessed 15 September 2023.

[250] R. Mullin. The Ethics of AI in the Lab. *C&EN Global Enterprise*, 101(31):30–35, 2023. doi:10.1021/cen-10131-cover.

[251] P. Murray-Rust. Chemistry for Everyone. *Nature*, 451(7179):648–651, 2008. doi:10.1038/451648a.

[252] P. Murray-Rust. CML - Frequently Asked Questions, 2012. URL http://www.xml-cml.org/documentation/FAQ.html#chemistry. Accessed 31 July 2021.

[253] P. Murray-Rust, H. S. Rzepa, S. M. Tyrrell, and Y. Zhang. Representation and Use of Chemistry in the Global Electronic Age. *Org. Biomol. Chem.*, 2(22):3192–3203, 2004. doi:10.1039/B410732B.

[254] M. A. Musen and Protégé Team. The Protégé Project: A Look Back and a Look Forward. *AI Matters*, 1(4):4–12, 2015. doi:10.1145/2757001.2757003.

[255] A. M. Nambiar, C. P. Breen, T. Hart, T. Kulesza, T. F. Jamison, and K. F. Jensen. Bayesian Optimization of Computer-Proposed Multistep Synthetic Routes on an Automated Robotic Flow Platform. *ACS Cent. Sci.*, 8(6):825–836, 2022. doi:10.1021/acscentsci.2c00207.

[256] NextMove Software. Pistachio, 2023. URL https://www.nextmovesoftware.com/pistachio.html. Accessed 17 July 2023.

[257] M. C. Nicklaus. NIH Virtual Workshop on Reaction Informatics, May 18-20, 2021. URL https://cactus.nci.nih.gov/presentations/NIHReactInf_2021-05/NIHReactInf.html. Accessed 31 July 2021.

[258] P. Nikolaev, D. Hooper, N. Perea-Lopez, M. Terrones, and B. Maruyama. Discovery of Wall-Selective Carbon Nanotube Growth Conditions via Automated Experimentation. *ACS Nano*, 8(10):10214–10222, 2014. doi:10.1021/nn503347a.

[259] P. Nikolaev, D. Hooper, F. Webber, R. Rao, K. Decker, M. Krein, J. Poleski, R. Barto, and B. Maruyama. Autonomy in Materials Research: A Case Study in Carbon Nanotube Growth. *npj Comput. Mater.*, 2:16031, 2016. doi:10.1038/npjcompumats.2016.31.

[260] M. M. Noack and J. A. Sethian. Autonomous Discovery in Science and Engineering, 8 2021. URL https://www.osti.gov/biblio/1818491. Accessed 14 November 2021.

[261] N. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor. Industry-Scale Knowledge Graphs: Lessons and Challenges. *Commun. ACM*, 62(8):36–43, 2019. doi:10.1145/3331166.

[262] N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison. Open Babel: An Open Chemical Toolbox. *J. Cheminf.*, 3:33, 2011. doi:10.1186/1758-2946-3-33.

[263] Office for National Statistics. ONS Geography Linked Data | SPARQL, 2023. URL https://statistics.data.gov.uk/sparql. Accessed 15 September 2023.

[264] R. S. Olson, R. J. Urbanowicz, P. C. Andrews, N. A. Lavender, L. C. Kidd, and J. H. Moore. Automating Biomedical Data Science Through Tree-Based Pipeline Optimization. In *European Conference on the Applications of Evolutionary Computation*, pages 123–137. Springer, 2016. doi:10.1007/978-3-319-31204-0_9.

[265] S. P. Ong, W. D. Richards, A. Jain, G. Hautier, M. Kocher, S. Cholia, D. Gunter, V. L. Chevrier, K. A. Persson, and G. Ceder. Python Materials Genomics (pymatgen): A Robust, Open-Source Python Library for Materials Analysis. *Comput. Mater. Sci.*, 68:314–319, 2013. doi:10.1016/j.commatsci.2012.10.028.

[266] ontotext. What is RDF-star?, 2022. URL https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-star/. Accessed 29 September 2023.

[267] OPC Foundation. Unified Architecture, 2023. URL https://opcfoundation.org/about/opc-technologies/opc-ua/. Accessed 8 March 2023.

[268] Open Reaction Database Project Authors. Welcome to the Open Reaction Database!, 2021. URL https://docs.open-reaction-database.org/en/latest/. Accessed 27 May 2021.

[269] OpenAI. GPT-4 Technical Report. *arXiv Preprint*, 2023. doi:10.48550/arXiv.2303.08774.

[270] S. O'Neill. AI-Driven Robotic Laboratories Show Promise. *Engineering*, 7(10): 1351–1353, 2021. doi:10.1016/j.eng.2021.08.006.

[271] M. Pan, J. Sikorski, C. A. Kastner, J. Akroyd, S. Mosbach, R. Lau, and M. Kraft. Applying Industry 4.0 to the Jurong Island Eco-industrial Park. *Energy Procedia*, 75: 1536–1541, 2015. doi:10.1016/j.egypro.2015.07.313.

[272] J. S. Park, P. Hu, Y. Lin, and L. A. Reinsalu. Composition and Method for Preventing, Reducing, Alleviating or Treating Idiopathic Vomiting, Feb. 4 2020. URL https://testpubchem.ncbi.nlm.nih.gov/patent/US-10548935-B2. US Patent 10,548,935. Accessed 11 February 2023.

[273] L. Pascazio, S. D. Rihm, A. Naseri, S. Mosbach, J. Akroyd, and M. Kraft. Chemical Species Ontology for Data Integration and Knowledge Discovery. *J. Chem. Inf. Model.*, 63(21):6569–6586, 2023. ISSN 1549-960X. doi:10.1021/acs.jcim.3c00820.

[274] N. Pattinson, M. Neidhardt, M. Hopkins, and H. Haas. Lab of the Future Panel. SLAS Europe 2023 Conference and Exhibition, Brussels, Belgium, 23-26 May, 2023. URL https://applied.slas.org/products/lab-of-the-future-panel#tab-product_tab_overview. Accessed 21 September 2023.

[275] B. G. Pelkie and L. D. Pozzo. The Laboratory of Babel: Highlighting Community Needs for Integrated Materials Data Management. *Digital Discovery*, 2(3):544–556, 2023. doi:10.1039/d3dd00022b.

[276] T. Pellissier Tanon, D. Vrandečić, S. Schaffert, T. Steiner, and L. Pintscher. From Freebase to Wikidata: The Great Migration. In *Proceedings of the 25th International Conference on World Wide Web*, pages 1419–1428, 2016. doi:10.1145/2872427.2874809.

[277] I. M. Pendleton, G. Cattabriga, Z. Li, M. A. Najeeb, S. A. Friedler, A. J. Norquist, E. M. Chan, and J. Schrier. Experiment Specification, Capture and Laboratory Automation Technology (ESCALATE): A Software Pipeline for Automated Chemical Experimentation and Data Management. *MRS Commun.*, 9(3):846–859, 2019. doi:10.1557/mrc.2019.72.

[278] M. Peplow. Organic Synthesis: The Robo-Chemist. *Nature*, 512(7512):20, 2014. doi:10.1038/512020a.

[279] W. Phadungsukanan, M. Kraft, J. A. Townsend, and P. Murray-Rust. The Semantics of Chemical Markup Language (CML) for Computational Chemistry: CompChem. *J. Cheminf.*, 4(1):15, 2012. doi:10.1186/1758-2946-4-15.

[280] Pistoia Alliance. Unified Data Model, 2020. URL https://github.com/PistoiaAlliance/UDM. Accessed 30 May 2023.

[281] Pistoia Alliance. Update from the Pistoia Alliance's Methods Hub Project, 2022. URL https://www.pistoiaalliance.org/methods/april-2022-methods-database-hplc-uv-methods/. Accessed 10 February 2023.

[282] A. Pomberger, A. A. P. McCarthy, A. Khan, S. Sung, C. J. Taylor, M. J. Gaunt, L. Colwell, D. Walz, and A. A. Lapkin. The effect of chemical representation on active machine learning towards closed-loop optimization. *React. Chem. Eng.*, 7(6): 1368–1379, 2022. doi:10.1039/d2re00008c.

[283] S. Poslad. Specifying Protocols for Multi-Agent Systems Interaction. *ACM Trans. Auton. Adapt. Syst.*, 2(4):15, 2007. doi:10.1145/1293731.1293735.

[284] E. O. Pyzer-Knapp, J. W. Pitera, P. W. Staar, S. Takeda, T. Laino, D. P. Sanders, J. Sexton, J. R. Smith, and A. Curioni. Accelerating Materials Discovery Using Artificial Intelligence, High Performance Computing and Robotics. *npj Comput. Mater.*, 8(1):84, 2022. doi:10.1038/s41524-022-00765-z.

[285] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, R. Tian, R. Xie, J. Zhou, M. Gerstein, D. Li, Z. Liu, and M. Sun. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-World APIs. *arXiv Preprint*, 2023. doi:10.48550/arXiv.2307.16789.

[286] H. Y. Quek, M. Hofmeister, S. D. Rihm, J. Yan, J. Lai, G. Brownbridge, M. Hillman, S. Mosbach, W. Ang, Y.-K. Tsai, D. N. Tran, S. Kang, W. Tan, and M. Kraft. BIM-GIS Integration: Knowledge Graphs in a World of Data Silos, 2023. Preprint at https://como.ceb.cam.ac.uk/preprints/311/.

[287] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. ROS: An Open-Source Robot Operating System. In *ICRA Workshop on Open Source Software*, volume 3. Kobe, Japan, 2009. URL http://robotics.stanford.edu/~ang/papers/icraoss09-ROS.pdf. Accessed 14 November 2021.

[288] P. Raccuglia, K. C. Elbert, P. D. F. Adler, C. Falk, M. B. Wenny, A. Mollo, M. Zeller, S. A. Friedler, J. Schrier, and A. J. Norquist. Machine-Learning-Assisted Materials Discovery Using Failed Experiments. *Nature*, 533(7601):73–76, 2016. doi:10.1038/nature17439.

[289] F. Rahmanian, J. Flowers, D. Guevarra, M. Richter, M. Fichtner, P. Donnely, J. M. Gregoire, and H. S. Stein. Enabling Modular Autonomous Feedback-Loops in Materials Science through Hierarchical Experimental Laboratory Automation and Orchestration. *Adv. Mater. Interfaces*, 9(8):2101987, 2022. doi:10.1002/admi.202101987.

[290] J. Ramírez, D. Soto, S. López, J. Akroyd, D. Nurkowski, M. L. Botero, N. Bianco, G. Brownbridge, M. Kraft, and A. Molina. A Virtual Laboratory to Support Chemical Reaction Engineering Courses Using Real-Life Problems and Industrial Software. *Educ. Chem. Eng.*, 33:36–44, 2020. doi:10.1016/j.ece.2020.07.002.

[291] A. S. Rao and M. P. Georgeff. Modeling Rational Agents within a BDI-Architecture. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, KR'91, page 473–484, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. ISBN 1558601651. doi:10.5555/3087158.3087205.

[292] A. S. Rao and M. P. Georgeff. BDI Agents: From Theory to Practice. In *Proceedings of the First International Conference on Multi-Agent Systems*, ICMAS-95, pages 312–319, 1995. URL https://cdn.aaai.org/ICMAS/1995/ICMAS95-042.pdf. Accessed 29 September 2023.

[293] N. Ravi, P. Chaturvedi, E. A. Huerta, Z. Liu, R. Chard, A. Scourtas, K. J. Schmidt, K. Chard, B. Blaiszik, and I. Foster. FAIR Principles for AI Models with a Practical Application for Accelerated High Energy Diffraction Microscopy. *Sci. Data*, 9(1), 2022. doi:10.1038/s41597-022-01712-9.

[294] S. Ren, Z. Wang, B. Li, H. Liu, and J. Wang. Development of a Reduced Polyoxymethylene Dimethyl Ethers (PODEn) Mechanism for Engine Applications. *Fuel*, 238:208–224, 2019. doi:10.1016/j.fuel.2018.10.111.

[295] Z. Ren, Z. Ren, Z. Zhang, T. Buonassisi, and J. Li. Autonomous Experiments Using Active Learning and AI. *Nat. Rev. Mater.*, 8(9):563–564, 2023. doi:10.1038/s41578-023-00588-4.

[296] Z. Ren, Z. Zhang, Y. Tian, and J. Li. CRESt – Copilot for Real-World Experimental Scientist. *ChemRxiv Preprint*, 2023. doi:10.26434/chemrxiv-2023-tnz1x.

[297] A. Reuther, C. Byun, W. Arcand, D. Bestor, B. Bergeron, M. Hubbell, M. Jones, P. Michaleas, A. Prout, A. Rosa, and J. Kepner. Scalable System Scheduling for HPC and Big Data. *J. Parallel Distrib. Comput.*, 111:76–92, 2018. doi:10.1016/j.jpdc.2017.06.009.

[298] S. D. Rihm, J. Bai, A. Kondinski, S. Mosbach, J. Akroyd, and M. Kraft. The Digital Lab Framework as Part of The World Avatar, 2023. Preprint at https://como.ceb.cam.ac.uk/preprints/314/.

[299] H. Rijgersberg, M. Van Assem, and J. Top. Ontology of Units of Measure and Related Concepts. *Semant. Web*, 4(1):3–13, 2013. doi:10.3233/SW-2012-0069.

[300] J. M. Roberts, M. F. Bean, S. R. Cole, W. K. Young, and H. E. Weston. Informatics in the Analytical Laboratory: Vision for a New Decade. *Am. Pharm. Rev.*, 13(6):60, 2010. URL https://www.americanpharmaceuticalreview.com/Featured-Articles/115071-Informatics-in-the-Analytical-Laboratory-Vision-for-a-New-Decade/.

[301] J. M. Roberts, M. F. Bean, S. R. Cole, W. K. Young, and H. E. Weston. The Adaptable Laboratory: A Holistic Informatics Architecture. *Am. Pharm. Rev.*, 14(1):12, 2011. URL https://www.americanpharmaceuticalreview.com/Featured-Articles/37098-The-Adaptable-Laboratory-A-Holistic-Informatics-Architecture/.

[302] L. M. Roch, F. Häse, C. Kreisbeck, T. Tamayo-Mendoza, L. P. E. Yunker, J. E. Hein, and A. Aspuru-Guzik. ChemOS: Orchestrating Autonomous Experimentation. *Sci. Robot.*, 3(19):eaat5559, 2018. doi:10.1126/scirobotics.aat5559.

[303] L. M. Roch, F. Häse, C. Kreisbeck, T. Tamayo-Mendoza, L. P. E. Yunker, J. E. Hein, and A. Aspuru-Guzik. ChemOS: An Orchestration Software to Democratize Autonomous Discovery. *PLoS One*, 15(4):e0229862, 2020. doi:10.1371/journal.pone.0229862.

[304] S. Rohrbach, M. Šiaučiulis, G. Chisholm, P.-A. Pirvan, M. Saleeb, S. H. M. Mehr, E. Trushina, A. I. Leonov, G. Keenan, A. Khan, A. Hammer, and L. Cronin. Digitization and Validation of a Chemical Synthesis Literature Database in the ChemPU. *Science*, 377(6602):172–180, 2022. doi:10.1126/science.abo0058.

[305] A. Rosen, S. Iyer, D. Ray, Z. Yao, A. Aspuru-Guzik, L. Gagliardi, J. Notestein, and R. Q. Snurr. Machine Learning the Quantum-Chemical Properties of Metal–Organic Frameworks for Accelerated Materials Discovery. *Matter*, 4(5):1578–1597, 2021. doi:10.1016/j.matt.2021.02.015.

[306] A. S. Rosen, J. M. Notestein, and R. Q. Snurr. Identifying Promising Metal–Organic Frameworks for Heterogeneous Catalysis via High-Throughput Periodic Density Functional Theory. *J. Comput. Chem.*, 40(12):1305–1318, 2019. doi:10.1002/jcc.25787.

[307] D. L. Roth. SPRESIweb 2.1, a Selective Chemical Synthesis and Reaction Database. *J. Chem. Inf. Model.*, 45(5):1470–1473, 2005. doi:10.1021/ci050274b.

[308] M. A. Rühl, R. Schäfer, and G. W. Kramer. Spectro ML-A Markup Language for Molecular Spectrometry Data. *JALA: J. Assoc. Lab. Autom.*, 6(6):76–82, 2001. doi:10.1016/S1535-5535-04-00168-6.

[309] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3rd edition, 2010.

[310] M. Sabou, S. Biffl, A. Einfalt, L. Krammer, W. Kastner, and F. J. Ekaputra. Semantics for Cyber-Physical Systems: A Cross-Domain Perspective. *Semant. Web*, 11(1):115–124, 2020. doi:10.3233/SW-190381. URL http://semantic-web-journal.net/content/semantics-cyber-physical-systems-cross-domain-perspective-0.

[311] L. M. Sanders, R. T. Scott, J. H. Yang, A. A. Qutub, H. G. Martin, D. C. Berrios, J. J. A. Hastings, J. Rask, G. Mackintosh, A. L. Hoarfrost, S. Chalk, J. Kalantari, K. Khezeli, E. L. Antonsen, J. Babdor, R. Barker, S. E. Baranzini, A. Beheshti, G. M. Delgado-Aparicio, B. S. Glicksberg, C. S. Greene, M. Haendel, A. A. Hamid, P. Heller, D. Jamieson, K. J. Jarvis, S. V. Komarova, M. Komorowski, P. Kothiyal, A. Mahabal, U. Manor, C. E. Mason, M. Matar, G. I. Mias, J. Miller, J. G. Myers, C. Nelson, J. Oribello, S. min Park, P. Parsons-Wingerter, R. K. Prabhu, R. J. Reynolds, A. Saravia-Butler, S. Saria, A. Sawyer, N. K. Singh, M. Snyder, F. Soboczenski, K. Soman, C. A. Theriot, D. V. Valen, K. Venkateswaran, L. Warren, L. Worthey, M. Zitnik, and S. V. Costes. Biological Research and Self-Driving Labs in Deep Space Supported by Artificial Intelligence. *Nat. Mach. Intell.*, 5(3):208–219, 2023. doi:10.1038/s42256-023-00618-4.

[312] T. Savage, J. Akroyd, S. Mosbach, M. Hillman, F. Sielker, and M. Kraft. Universal Digital Twin–The Impact of Heat Pumps on Social Inequality. *Adv. Appl. Energy*, 5:100079, 2022. doi:10.1016/j.adapen.2021.100079.

[313] B. Schäfer. Data Exchange in the Laboratory of the Future – A Glimpse at AnIML and SiLA, 2018. URL https://analyticalscience.wiley.com/do/10.1002/gitlab.17270/full/. Accessed 30 May 2023.

[314] K. Schekotihin, P. Rodler, and W. Schmid. OntoDebug: Interactive Ontology Debugging Plug-In for Protégé. In *Lecture Notes in Computer Science*, pages 340–359. Springer International Publishing, 2018. doi:10.1007/978-3-319-90050-6_19.

[315] G. Schneider. Automating Drug Discovery. *Nat. Rev. Drug Discov.*, 17(2):97–113, 2018. doi:10.1038/nrd.2017.232.

[316] N. Schneider, D. M. Lowe, R. A. Sayle, M. A. Tarselli, and G. A. Landrum. Big Data from Pharmaceutical Patents: A Computational Analysis of Medicinal Chemists' Bread and Butter. *J. Med. Chem.*, 59(9):4385–4402, 2016. doi:10.1021/acs.jmedchem.6b00153.

[317] G. Schreiber and Y. Raimond. RDF 1.1 Primer. W3C Working Group Note 24 June 2014, 2014. URL https://www.w3.org/TR/rdf11-primer/. Accessed 13 September 2023.

[318] S. Schulz, B. Suntisrivaraporn, F. Baader, and M. Boeker. SNOMED Reaching its Adolescence: Ontologists' and Logicians' Health Check. *Int. J. Med. Inform.*, 78: S86–S94, 2009. doi:10.1016/j.ijmedinf.2008.06.004.

[319] P. Schwaller, T. Gaudin, D. Lanyi, C. Bekas, and T. Laino. "Found in Translation": Predicting Outcomes of Complex Organic Chemistry Reactions Using Neural Sequence-to-Sequence Models. *Chem. Sci.*, 9(28):6091–6098, 2018. doi:10.1039/C8SC02339E.

[320] P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas, and A. A. Lee. Molecular Transformer: A Model for Uncertainty-Calibrated Chemical Reaction Prediction. *ACS Cent. Sci.*, 5(9):1572–1583, 2019. doi:10.1021/acscentsci.9b00576.

[321] P. Schwaller, R. Petraglia, V. Zullo, V. H. Nair, R. A. Haeuselmann, R. Pisoni, C. Bekas, A. Iuliano, and T. Laino. Predicting Retrosynthetic Pathways Using Transformer-Based Models and a Hyper-Graph Exploration Strategy. *Chem. Sci.*, 11(12):3316–3325, 2020. doi:10.1039/C9SC05704H.

[322] P. Schwaller, B. Hoover, J.-L. Reymond, H. Strobelt, and T. Laino. Extraction of Organic Chemistry Grammar from Unsupervised Learning of Chemical Reactions. *Sci. Adv.*, 7(15):eabe4166, 2021. doi:10.1126/sciadv.abe4166.

[323] A. M. Schweidtmann, A. D. Clayton, N. Holmes, E. Bradford, R. A. Bourne, and A. A. Lapkin. Machine Learning Meets Continuous Flow Chemistry: Automated Optimization Towards the Pareto Front of Multiple Objectives. *Chem. Eng. J.*, 352: 277–282, 2018. doi:10.1016/j.cej.2018.07.031.

[324] M. H. S. Segler and M. P. Waller. Modelling Chemical Reasoning to Predict and Invent Reactions. *Chem. Eur. J.*, 23(25):6118–6128, 2017. doi:10.1002/chem.201604556.

[325] M. H. S. Segler, M. Preuss, and M. P. Waller. Planning Chemical Syntheses with Deep Neural Networks and Symbolic AI. *Nature*, 555(7698):604–610, 2018. doi:10.1038/nature25978.

[326] M. Seifrid, J. Hattrick-Simpers, A. Aspuru-Guzik, T. Kalil, and S. Cranford. Reaching Critical MASS: Crowdsourcing Designs for the Next Generation of Materials Acceleration Platforms. *Matter*, 5(7):1972–1976, 2022. doi:10.1016/j.matt.2022.05.035.

[327] M. Seifrid, R. Pollice, A. Aguilar-Granda, Z. Morgan Chan, K. Hotta, C. T. Ser, J. Vestfrid, T. C. Wu, and A. Aspuru-Guzik. Autonomous Chemical Experiments: Challenges and Perspectives on Establishing a Self-Driving Lab. *Acc. Chem. Res.*, 55 (17):2454–2466, 2022. doi:acs.accounts.2c00220.

[328] D. A. Sheen and H. Wang. The Method of Uncertainty Quantification and Minimization Using Polynomial Chaos Expansions. *Combust. Flame*, 158(12):2358–2374, 2011. doi:10.1016/j.combustflame.2011.05.010.

[329] B. J. Shields, J. Stevens, J. Li, M. Parasram, F. Damani, J. I. M. Alvarado, J. M. Janey, R. P. Adams, and A. G. Doyle. Bayesian Reaction Optimization as a Tool for Chemical Synthesis. *Nature*, 590(7844):89–96, 2021. doi:10.1038/s41586-021-03213-y.

[330] J. J. Sikorski, G. Brownbridge, S. S. Garud, S. Mosbach, I. A. Karimi, and M. Kraft. Parameterisation of a Biodiesel Plant Process Flow Sheet Model. *Comput. Chem. Eng.*, 95:108–122, 2016. doi:10.1016/j.compchemeng.2016.06.019.

[331] L. F. Sikos and D. Philp. Provenance-Aware Knowledge Representation: A Survey of Data Models and Contextualized Knowledge Graphs. *Data Sci. and Eng.*, 5(3): 293–316, 2020. doi:10.1007/s41019-020-00118-0.

[332] SiLA. SiLA Rapid Integration | Standardization in Lab Automation, 2021. URL https://sila-standard.com/. Accessed 27 May 2021.

[333] M. Sim, M. G. Vakili, F. Strieth-Kalthoff, H. Hao, R. Hickman, S. Miret, S. Pablo-García, and A. Aspuru-Guzik. ChemOS 2.0: An Orchestration Architecture for Chemical Self-Driving Laboratories. *ChemRxiv Preprint*, 2023. doi:10.26434/chemrxiv-2023-v2khf.

[334] A. Singhal. Introducing the Knowledge Graph: Things, not Strings, 2012. URL https://blog.google/products/search/introducing-knowledge-graph-things-not/. Accessed 16 September 2023.

[335] R. A. Skilton, R. A. Bourne, Z. Amara, R. Horvath, J. Jin, M. J. Scully, E. Streng, S. L. Y. Tang, P. A. Summers, J. Wang, E. Pérez, N. Asfaw, G. L. P. Aydos, J. Dupont, G. Comak, M. W. George, and M. Poliakoff. Remote-Controlled Experiments with Cloud Chemistry. *Nat. Chem.*, 7(1):1–5, 2015. doi:10.1038/nchem.2143.

[336] M. Skreta, N. Yoshikawa, S. Arellano-Rubach, Z. Ji, L. B. Kristensen, K. Darvish, A. Aspuru-Guzik, F. Shkurti, and A. Garg. Errors are Useful Prompts: Instruction Guided Task Programming with Verifier-Assisted Iterative Prompting. *arXiv Preprint*, 2023. doi:10.48550/arXiv.2303.14100.

[337] I. M. Sobol. On the Systematic Search in a Hypercube. *SIAM J. Numer. Anal.*, 16(5): 790–793, 1979. doi:10.1137/070709359.

[338] E. Soedarmadji, H. S. Stein, S. K. Suram, D. Guevarra, and J. M. Gregoire. Tracking Materials Science Data Lineage to Manage Millions of Materials Experiments and Analyses. *npj Comput. Mater.*, 5:79, 2019. doi:10.1038/s41524-019-0216-x.

[339] L. N. Soldatova and R. D. King. An Ontology of Scientific Experiments. *J. R. Soc., Interface*, 3(11):795–803, 2006. doi:10.1098/rsif.2006.0134.

[340] M. Sporny, D. Longley, G. Kellogg, M. Lanthaler, P.-A. Champin, and N. Lindström. JSON-LD 1.1 – A JSON-based Serialization for Linked Data. W3C Recommendation 16 July 2020, 2020. URL https://www.w3.org/TR/json-ld11/. Accessed 16 September 2023.

[341] S. Staab and R. Studer. *Handbook on Ontologies*. Springer Science & Business Media, 2010.

[342] E. Stach, B. DeCost, A. G. Kusne, J. Hattrick-Simpers, K. A. Brown, K. G. Reyes, J. Schrier, S. Billinge, T. Buonassisi, I. Foster, C. P. Gomes, J. M. Gregoire, A. Mehta, J. Montoya, E. Olivetti, C. Park, E. Rotenberg, S. K. Saikin, S. Smullin, V. Stanev, and B. Maruyama. Autonomous Experimentation Systems for Materials Development: A Community Perspective. *Matter*, 4(9):2702–2726, 2021. doi:10.1016/j.matt.2021.06.036.

[343] M. J. Statt, B. A. Rohr, K. Brown, D. Guevarra, J. Hummelshøj, L. Hung, A. Anapolsky, J. M. Gregoire, and S. K. Suram. ESAMP: Event-Sourced Architecture for Materials Provenance Management and Application to Accelerated Materials Discovery. *Digital Discovery*, 2(4):1078–1088, 2023. doi:10.1039/d3dd00054k.

[344] M. J. Statt, B. A. Rohr, D. Guevarra, J. Breeden, S. K. Suram, and J. M. Gregoire. The Materials Experiment Knowledge Graph. *Digital Discovery*, 2(4):909–914, 2023. doi:10.1039/d3dd00067b.

[345] H. S. Stein and J. M. Gregoire. Progress and Prospects for Accelerating Materials Science with Automated and Autonomous Workflows. *Chem. Sci.*, 10(42):9640–9649, 2019. doi:10.1039/C9SC03766G.

[346] H. S. Stein, A. Sanin, F. Rahmanian, B. Zhang, M. Vogler, J. K. Flowers, L. Fischer, S. Fuchs, N. Choudhary, and L. Schroeder. From Materials Discovery to System Optimization by Integrating Combinatorial Electrochemistry and Data Science. *Curr. Opin. Electrochem.*, page 101053, 2022. doi:10.1016/j.coelec.2022.101053.

[347] S. Steiner, J. Wolf, S. Glatzel, A. Andreou, J. M. Granda, G. Keenan, T. Hinkley, G. Aragon-Camarasa, P. J. Kitson, D. Angelone, and L. Cronin. Organic Synthesis in a Modular Robotic System Driven by a Chemical Programming Language. *Science*, 363(6423):eaav2211, 2019. doi:10.1126/science.aav2211.

[348] W. Sun, G. Wang, S. Li, R. Zhang, B. Yang, J. Yang, Y. Li, C. K. Westbrook, and C. K. Law. Speciation and the Laminar Burning Velocities of Poly(Oxymethylene) Dimethyl Ether 3 (POMDME3) Flames: An Experimental and Modeling Study. *Proc. Combust. Inst.*, 36(1):1269–1278, 2017. doi:10.1016/j.proci.2016.05.058.

[349] M. C. Swain and J. M. Cole. ChemDataExtractor: A Toolkit for Automated Extraction of Chemical Information from the Scientific Literature. *J. Chem. Inf. Model.*, 56(10): 1894–1904, 2016. doi:10.1021/acs.jcim.6b00207.

[350] D. P. Tabor, L. M. Roch, S. K. Saikin, C. Kreisbeck, D. Sheberla, J. H. Montoya, S. Dwaraknath, M. Aykol, C. Ortiz, H. Tribukait, C. Amador-Bedolla, C. J. Brabec, B. Maruyama, K. A. Persson, and A. Aspuru-Guzik. Accelerating the Discovery of Materials for Clean Energy in the Era of Smart Automation. *Nat. Rev. Mater.*, 3(5): 5–20, 2018. doi:10.1038/s41578-018-0005-z.

[351] M. J. Tamasi and A. J. Gormley. Biologic Formulation in a Self-Driving Biomaterials Lab. *Cell Rep. Phys. Sci.*, 3(9):101041, 2022. doi:10.1016/j.xcrp.2022.101041.

[352] F. Tao and Q. Qi. Make More Digital Twins. *Nature*, 573:490–491, 2019. doi:10.1038/d41586-019-02849-1.

[353] C. J. Taylor, M. Booth, J. A. Manson, M. J. Willis, G. Clemens, B. A. Taylor, T. W. Chamberlain, and R. A. Bourne. Rapid, Automated Determination of Reaction Models and Kinetic Parameters. *Chem. Eng. J.*, 413:127017, 2021. doi:10.1016/j.cej.2020.127017.

[354] C. J. Taylor, A. Pomberger, K. C. Felton, R. Grainger, M. Barecka, T. W. Chamberlain, R. A. Bourne, C. N. Johnson, and A. A. Lapkin. A Brief Introduction to Chemical Reaction Optimization. *Chem. Rev.*, 123(6):3089–3126, 2023. doi:10.1021/acs.chemrev.2c00798.

[355] A. Thakkar, S. Johansson, K. Jorner, D. Buttar, J.-L. Reymond, and O. Engkvist. Artificial Intelligence and Automation in Computer Aided Synthesis Planning. *React. Chem. Eng.*, 6(1):27–51, 2021. doi:10.1039/D0RE00340A.

[356] The Apache Software Foundation. Apache Airflow, 2022. URL https://airflow.apache.org/. Accessed 17 July 2022.

[357] The Foundation for Intelligent Physical Agents. Welcome to the Foundation for Intelligent Physical Agents, 2020. URL http://www.fipa.org/. Accessed 14 July 2022.

[358] The Gene Ontology Consortium. The Gene Ontology Project in 2008. *Nucleic Acids Res.*, 36(suppl_1):D440–D444, 2008. doi:10.1093/nar/gkm883.

[359] The W3C SPARQL Working Group (Ed.). SPARQL 1.1 Overview. W3C Recommendation 21 March 2013, 2013. URL https://www.w3.org/TR/sparql11-overview/. Accessed 13 September 2023.

[360] Thermo Fisher Scientific (Informatics). An XML-Based File Format for Archival Storage of Analytical Instrument Data, 2001. URL http://www.gaml.org/Documentation/XML%20Analytical%20Archive%20Format.pdf. Accessed 31 July 2021.

[361] K. Tran and Z. W. Ulissi. Active Learning Across Intermetallics to Guide Discovery of Electrocatalysts for CO_2 Reduction and H_2 Evolution. *Nat. Catal.*, 1(9):696–703, 2018. doi:10.1038/s41929-018-0142-1.

[362] K. Tran, A. Palizhati, S. Back, and Z. W. Ulissi. Dynamic Workflows for Routine Materials Discovery in Surface Science. *J. Chem. Inf. Model.*, 58(12):2392–2400, 2018. doi:10.1021/acs.jcim.8b00386.

[363] P. Tremouilhac, C.-L. Lin, P.-C. Huang, Y.-C. Huang, A. Nguyen, N. Jung, F. Bach, R. Ulrich, B. Neumair, A. Streit, and S. Bräse. The Repository Chemotion: Infrastructure for Sustainable Research in Chemistry. *Angew. Chem., Int. Ed.*, 59(50): 22771–22778, 2020. doi:10.1002/anie.202007702.

[364] T. Turányi and H. Rabitz. Local methods. In A. Saltelli, K. Chan, and E. M. Scott, editors, *Sensitivity Analysis*, Wiley Series in Probability and Statistics, pages 81–99. John Wiley & Sons, New York, 2000.

[365] U.S. Government. Data Catalog, 2023. URL https://catalog.data.gov/dataset. Accessed 14 September 2023.

[366] T. Varga, T. Turányi, E. Czinki, T. Furtenbacher, and A. Császár. ReSpecTh: A Joint Reaction Kinetics, Spectroscopy, and Thermochemistry Information System. In *Proc. of the 7th Eur. Combust. Meet.*, volume 30, pages 1–5. Budapest, Hungary, 2015.

[367] T. Varga, Á. Busai, I. G. Zsély, T. Nagy, and T. Turányi. Optima++ v1. 2: A General C++ Framework for Performing Combustion Simulations and Mechanism Optimization, 2020. URL http://respecth.hu/.

[368] J. J. Varghese, L. Cao, C. Robertson, Y. Yang, L. F. Gladden, A. A. Lapkin, and S. H. Mushrif. Synergistic Contribution of the Acidic Metal Oxide–Metal Couple and Solvent Environment in the Selective Hydrogenolysis of Glycerol: A Combined Experimental and Computational Study Using ReO_X–Ir as the Catalyst. *ACS Catal.*, 9 (1):485–503, 2019. doi:10.1021/acscatal.8b03079.

[369] A. C. Vaucher, F. Zipoli, J. Geluykens, V. H. Nair, P. Schwaller, and T. Laino. Automated Extraction of Chemical Synthesis Actions from Experimental Procedures. *Nat. Commun.*, 11:3601, 2020. doi:10.1038/s41467-020-17266-6.

[370] T. Vieira, A. C. Stevens, A. Chtchemelinine, D. Gao, P. Badalov, and L. Heumann. Development of a Large-Scale Cyanation Process Using Continuous Flow Chemistry En Route to the Synthesis of Remdesivir. *Org. Process Res. Dev.*, 24(10):2113–2121, 2020. doi:10.1021/acs.oprd.0c00172.

[371] A. A. Volk, R. W. Epps, D. T. Yonemoto, B. S. Masters, F. N. Castellano, K. G. Reyes, and M. Abolhasani. AlphaFlow: Autonomous Discovery and Optimization of Multi-Step Chemistry Using a Self-Driven Fluidic Lab Guided by Reinforcement Learning. *Nat. Commun.*, 14(1), 2023. doi:10.1038/s41467-023-37139-y.

[372] L. von Rueden, S. Mayer, K. Beckh, B. Georgiev, S. Giesselbach, R. Heese, B. Kirsch, M. Walczak, J. Pfrommer, A. Pick, R. Ramamurthy, J. Garcke, C. Bauckhage, and J. Schuecker. Informed Machine Learning - A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems. *IEEE Trans. Knowl. Data Eng.*, 35(1): 614–633, 2021. doi:10.1109/TKDE.2021.3079836.

[373] A. Vriza, H. Chan, and J. Xu. Self-Driving Laboratory for Polymer Electronics. *Chem. Mater.*, 35(8):3046–3056, 2023. doi:10.1021/acs.chemmater.2c03593.

[374] W3C. Semantic Web, 2015. URL https://www.w3.org/standards/semanticweb/. Accessed 1 June 2021.

[375] W3C OWL Working Group (Ed.). OWL 2 Web Ontology Language Document Overview (Second Edition). W3C Recommendation 11 December 2012, 2012. URL https://www.w3.org/TR/owl2-overview/. Accessed 13 September 2023.

[376] C. Waldron, A. Pankajakshan, M. Quaglio, E. Cao, F. Galvanin, and A. Gavriilidis. An Autonomous Microreactor Platform for the Rapid Identification of Kinetic Models. *React. Chem. Eng.*, 4(9):1623–1636, 2019. doi:10.1039/C8RE00345A.

[377] J. Warnatz, U. Maas, and R. W. Dibble. *Combustion: Physical and Chemical Fundamentals, Modeling and Simulation, Experiments, Pollutant Formation*. Springer, Berlin, 2006.

[378] B. W. Weber and K. E. Niemeyer. ChemKED: A Human- and Machine-Readable Data Standard for Chemical Kinetics Experiments. *Int. J. Chem. Kinet.*, 50(3):135–148, 2018. doi:10.1002/kin.21142.

[379] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv Preprint*, 2022. doi:10.48550/arXiv.2201.11903.

[380] D. Weininger. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.*, 28(1):31–36, 1988. doi:10.1021/ci00057a005.

[381] P. B. Wigley, P. J. Everitt, A. van den Hengel, J. W. Bastian, M. A. Sooriyabandara, G. D. McDonald, K. S. Hardman, C. D. Quinlivan, P. Manju, C. C. N. Kuhn, I. R. Petersen, A. N. Luiten, J. J. Hope, N. P. Robins, and M. R. Hush. Fast Machine-Learning Online Optimization of Ultra-Cold-Atom Experiments. *Sci. Rep.*, 6:25890, 2016. doi:10.1038/srep25890.

[382] L. Wilbraham, S. H. M. Mehr, and L. Cronin. Digitizing Chemistry Using the Chemical Processing Unit: From Synthesis to Discovery. *Acc. Chem. Res.*, 54(2): 253–262, 2021. doi:10.1021/acs.accounts.0c00674.

[383] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons. The FAIR Guiding Principles for Scientific Data Management and Stewardship. *Sci. Data*, 3:160018, 2016. doi:10.1038/sdata.2016.18.

[384] E. L. Willighagen, A. Waagmeester, O. Spjuth, P. Ansell, A. J. Williams, V. Tkachenko, J. Hastings, B. Chen, and D. J. Wild. The ChEMBL Database as Linked Open Data. *J. Cheminf.*, 5:23, 2013. doi:10.1186/1758-2946-5-23.

[385] H. Winicov, J. Schainbaum, J. Buckley, G. Longino, J. Hill, and C. Berkoff. Chemical Process Optimization by Computer—A Self-Directed Chemical Synthesis System. *Anal. Chim. Acta*, 103(4):469–476, 1978. doi:10.1016/S0003-2670(01)83110-X.

[386] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics, 2020.

doi:10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

[387] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2009.

[388] M. Wooldridge. What Is Missing from Contemporary AI? The World. *Intell. Comput.*, 2022, 2022. doi:10.34133/2022/9847630.

[389] World Wide Web Consortium (W3C). RDF-star, 2021. URL https://w3c.github.io/rdf-star/.

[390] D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue, and T. Lookman. Accelerated Search for Materials with Targeted Properties by Adaptive Design. *Nat. Commun.*, 7(1):1–9, 2016. doi:10.1038/ncomms11241.

[391] N. Yachie and T. Natsume. Robotic Crowd Biology with Maholo LabDroids. *Nat. Biotechnol.*, 35(4):310–312, 2017. doi:10.1038/nbt.3758.

[392] H. Yang, J. Li, K. Z. Lim, C. Pan, T. Van Truong, Q. Wang, K. Li, S. Li, X. Xiao, M. Ding, T. Chen, X. Liu, Q. Xie, P. V. y. Alvarado, X. Wang, and P.-Y. Chen. Automatic Strain Sensor Design via Active Learning and Data Augmentation for Soft Machines. *Nat. Mach. Intell.*, 4(1):84–94, 2022. doi:10.1038/s42256-021-00434-8.

[393] A. B. Yoo, M. A. Jette, and M. Grondona. SLURM: Simple Linux Utility for Resource Management. In *Job Scheduling Strategies for Parallel Processing*, pages 44–60. Springer Berlin Heidelberg, 2003. doi:10.1007/10968987_3.

[394] C. Yu, M. Seslija, G. Brownbridge, S. Mosbach, M. Kraft, M. Parsi, M. Davis, V. Page, and A. Bhave. Deep Kernel Learning Approach to Engine Emissions Modeling. *Data-Centric Eng.*, 1:e4, 2020. doi:10.1017/dce.2020.4.

[395] J. Yu and R. Buyya. A Taxonomy of Workflow Management Systems for Grid Computing. *J. Grid Comput.*, 3(3):171–200, 2005. doi:10.1007/s10723-005-9010-8.

[396] C. Zhang, A. Romagnoli, L. Zhou, and M. Kraft. Knowledge Management of Eco-industrial Park for Efficient Energy Utilization Through Ontology-Based Approach. *Appl. Energy*, 204:1412–1421, 2017. doi:10.1016/j.apenergy.2017.03.130.

[397] J. Zhao, C. Bizer, Y. Gil, P. Missier, and S. Sahoo. Provenance Requirements for the Next Version of RDF. W3C Workshop – RDF Next Steps, Stanford, CA, USA, June 26-27, 2010. URL https://www.w3.org/2009/12/rdf-ws/papers/ws08. Accessed 29 September 2023.

[398] Z. Zheng, O. Zhang, C. Borgs, J. T. Chayes, and O. M. Yaghi. ChatGPT Chemistry Assistant for Text Mining and the Prediction of MOF Synthesis. *J. Am. Chem. Soc.*, 145(32):18048–18062, 2023. doi:10.1021/jacs.3c05819.

[399] L. Zhou, M. Pan, J. J. Sikorski, S. Garud, L. K. Aditya, M. J. Kleinelanghorst, I. A. Karimi, and M. Kraft. Towards an Ontological Infrastructure for Chemical Process Simulation and Optimization in the Context of Eco-industrial Parks. *Appl. Energy*, 204:1284–1298, 2017. doi:10.1016/j.apenergy.2017.05.002.

[400] Q. Zhou, P. Tang, S. Liu, J. Pan, Q. Yan, and S.-C. Zhang. Learning Atoms for Materials Discovery. *Proc. Natl. Acad. Sci. U. S. A.*, 115(28):E6411–E6417, 2018. doi:10.1073/pnas.1801181115.

[401] X. Zhou, A. Eibeck, M. Q. Lim, N. B. Krdzavac, and M. Kraft. An Agent Composition Framework for the J-Park Simulator - A Knowledge Graph for the Process Industry. *Comput. Chem. Eng.*, 130:106577, 2019. doi:10.1016/j.compchemeng.2019.106577.

[402] X. Zhou, M. Q. Lim, and M. Kraft. A Smart Contract-based Agent Marketplace for the J-Park Simulator - A Knowledge Graph for the Process Industry. *Comput. Chem. Eng.*, 139:106896, 2020. doi:10.1016/j.compchemeng.2020.106896.

[403] Q. Zhu, F. Zhang, Y. Huang, H. Xiao, L. Zhao, X. Zhang, T. Song, X. Tang, X. Li, G. He, B. Chong, J. Zhou, Y. Zhang, B. Zhang, J. Cao, M. Luo, S. Wang, G. Ye, W. Zhang, X. Chen, S. Cong, D. Zhou, H. Li, J. Li, G. Zou, W. Shang, J. Jiang, and Y. Luo. An All-Round AI-Chemist with a Scientific Mind. *Natl. Sci. Rev.*, 9(10), 2022. doi:10.1093/nsr/nwac190.

Appendix A

Data infrastructure in contemporary self-driving laboratories

This appendix lists the detailed findings from the selected state-of-the-art studies in self-driving laboratories (SDLs). To the best of our knowledge, we identified the functional component realisation in a platform-based approach in Table A.1. Besides, in Table A.2, we categorised the data flow and communication protocols between the functional components following the method described in the main text.

A.1 Realisation of functional components

Table A.1 Functional component realisation of selected state-of-the-art studies in SDLs. For computational model development applications, the model generated as planner was trained on executor with user-defined optimisers. The executors are physical hardware unless otherwise stated. HPC: high-performance computing. MS: mass spectrometer. IR: infrared spectroscopy. BPR: back pressure regulator. HPLC: high-performance liquid chromatography. NMD-M3: Nanotechnology Materials Data Mining, Modeling & Management. VASP: Vienna *ab initio* Simulation Package. ASE: Atomic Simulation Environment. ICSD: Inorganic Crystal Structure Database.

Reference	**Application**	**System**	**Receptionist**	**Coordinator**	**Librarian**	**Planner**	**Executor**
Fitzpatrick et al. [105]	Reaction condition optimisation of a three-dimensional heterogeneous catalytic reaction and a five-dimensional Appel reaction	Flow reactor	Web browser with credentials	LeyLab: PHP-based	Database	Complex Method [182]	Vapourtec R2/R4, online MS, webcam, and inline IR
Ingham et al. [172]	Multi-step synthesis of 2-aminoadamantane-2-carboxylic acid	Flow reactor	Web browser	Octopus: Python-based	CSV file	N/A	Uniqsis, Vapourtec R2+, Knauer machine (HPLC pump), and webcams
Fitzpatrick et al. [106]	Reaction condition optimisation for three active pharmaceutical ingredients (APIs)	Flow reactor	Web browser	LeyLab: PHP-based	Database: MySQL	Complex Method [182]	Flow reactor, HPLC, inline IR, BPR, computer vision module, *etc.*
Nikolaev et al. [259]	Automated designing, executing, and evaluating carbon nanotube growth experiments	Batch reactor	NMD-M3 software	NMD-M3 software	Database within NMD-M3 software	Random forest and genetic algorithm	In-house built ARES platform: inverted Raman microscope, laser, pressure gauge, vaccum pump and gas mass glow controller
Wigley et al. [381]	Process condition optimisation of the production of Bose-Einstein condensates (BEC)	Exponential evaporation ramp	N/A	Python code	MAT file	MLOO [381]: Gaussian process-based	Cooling ramp, evaporation process
Greenaway et al. [133]	Automated porous organic cages discovery by high-throughput screening	Batch reactor	Chemspeed software	Python code	CSV from Chemspeed	Predefined algorithmic workflow in Fig. 6 in Greenaway et al. [133]	**Computational:** HPC; **Physical:** Chemspeed Accelerator SLT-100 automated synthesis platform, NMR, HRMS, HPLC, *etc.*

Table A.1 (Continued)

Reference	Application	System	Receptionist	Coordinator	Librarian	Planner	Executor
Bédard et al. [17]	High-yielding implementations of C-C and C-N cross-coupling, olefination, reductive amination, nucleophilic aromatic substitution (S_NAr), photoredox catalysis, and a multistep sequence	Flow reactor	LabVIEW-based GUI	Matlab code	MAT file	SNOBFIT algorithm	Reactor, pump, pressure sensor, flow meter, phase sensor, IR based temperature sensor, camera, HPLC, IR, raman spectroscopy, MS
Caramelli et al. [43]	Collaborative chemical space exploration by two internet-connected robots on multiple chemical processes	Batch reactor	N/A	Python code	Twitter feed	Random search, grid search, and Monte-Carlo	Peristaltic pumps, webcam, and reaction flask
Skilton et al. [335]	Etherification of n-propanol in supercritical CO_2 over a γ-Al_2O_3 catalyst, optimized for the formation of di-n-propyl ether	Flow reactor	GLC Solutions	GLC Solutions	Stored within GLC Solutions	GLC Solutions software	Require local human operator for filling stock
Coley et al. [59]	Automated synthesis of 15 medicinal relevant small molecules	Flow reactor	Web browser (for ASKCOS) and Python-based GUI (for robotic execution)	Human researcher	Database (USPTO and Reaxys)	ASKCOS package askcos.mit.edu	Modularised flow reactor, six axis UR3 Universal Robot, BPR, MS, HPLC, NMR, *etc.*
Roch et al. [303]	Automated chemical recipe discovery	Batch reactor	NLP-based chatbot (Twitter, Slack, Gmail)	ChemOS: Python-based	SQLite database	Random search, Spearmint, SMAC, and Phoenics [303]	Various lab equipment based on user needs: pumps, HPLC, *etc.*
Montoya et al. [241]	End-to-end computational system for autonomous materials discovery	Binary/ternary inorganic chemicals	N/A	Python code	JSON	Algorithms in Fig. 2 in Montoya et al. [241]	**Computational:** DFT simulation on AWS EC2
Li et al. [209]	Discovery of optically active chiral inorganic perovskite nanocrystals	Flow reactor	Web browser with SSH login and Python-based GUI	MAOSIC: Python-based	DBMS, *e.g.*, MySQL	SNOBFIT	Microfludic reactor, collaborative robot, syringe pump, environment sensor, *etc.*

Table A.1 (Continued)

Reference	Application	System	Receptionist	Coordinator	Librarian	Planner	Executor
Xue et al. [390]	Materials discovery for very low thermal hysteresis (ΔT) multicomponent NiTi-based shape memory alloys	Batch reactor	N/A	Python code	CSV (based on supplementary information)	Efficient global optimisation and knowledge gradient	**Computational:** QUANTUM ESPRESSO planewave pseudopotential package; **Physical:** synthesis performed manually
Cao et al. [41]	Formulated product recipe optimisation	Batch reactor	N/A	Python code	CSV and textfile	TSEMO [27]	Robotic platform, syringe pumps, pH analyser, turbidity analyser, and viscometer
Jeraal et al. [178]	Multi-objective reaction condition optimisation for aldol condensation reaction	Flow reactor	Matlab-based GUI	Matlab application	CSV file	TSEMO [27]	Vapourtec R2/R4, Agilent 1260 HPLC, and BPR
King et al. [190]	Identification of genes encoding orphan enzymes in yeast *Saccharomyces cerevisiae*	Mobile robotic platform	N/A	Adam: robot scientist	KEGG (prior), MySQL database (new)	Bioinformatic methods (hypotheses generation); two-factor DoE	Freezer, liquid handler, incubators, *etc.*
King et al. [189]	Functional genomics synthesis optimisation for aromatic amino acid synthesis pathway in yeast	Mobile robotic platform	N/A	Laboratory Information Management System	Database within LIMS	Bayesian analysis of decision-tree learning	Liquid handling, pipetting and mixing liquids on microtitre plates
Ingham et al. [171]	Multi-step reaction condition optimisation for pyrazine-2-carboxamide and piperazine-2-carboxamide	Flow reactor	Web browser	Octopus: Python-based	CSV file	Complex Method [182]	Vapourtec R2+/R4, HPLC, computer vision module, *etc.*
Schweidtmann et al. [323]	Multi-objective reaction condition optimisation for S_NAr reaction and N-benzylation	Flow reactor	N/A	Matlab code	CSV file	TSEMO [27]	JASCO PU980 pumps, Vapourtec, Agilent 1100 HPLC, and BPR
HamediRad et al. [142]	Biosynthetic pathway optimisation of lycopene	Continuous workflow	N/A	BioAutomata: Python-based	DAT file	Bayesian optimisation	iBioFAB automated platform [47]
MacLeod et al. [221]	Self-driving laboratory for accelerated discovery of thin-film materials	Mobile robotic platform	Python-based GUI	ChemOS: Python-based	Database	Phoenics within ChemOS	North Robotics N9 robots and liquid handler

Table A.1 (Continued)

Reference	Application	System	Receptionist	Coordinator	Librarian	Planner	Executor
Segler et al. [325]	Computational model development for synthesis planning of small molecules	Single-step reactions	N/A	Python code	Extracted from ZINC and Reaxys	Neural network combined with Monte Carlo tree search	**Computational:** single NVIDIA K80 graphics processing unit
Steiner et al. [347]	Automated synthesis of three pharmaceutical compounds: diphenhydramine hydrochloride, rufinamide, and sildenafil	Batch reactor	N/A	Chemputer: Python-based	N/A	Synthesis route set by human	Reactor, pumps, filter, separator, and rotary evaporator
Mehr et al. [229]	Automated syntheses of 12 compounds from the literature, including the analgesic lidocaine, the Dess-Martin periodinane oxidation reagent, and the fluorinating agent AlkylFluor	Batch reactor	Web browser (ChemIDE), allow editing proposed synthesis steps	Chemputer: Python-based	Literature operation procedure in textfile (free-text format)	SynthReader: NLP-based synthesis action sequence planner	Reactor, pumps, filter, separator, rotary evaporator, vacuum, stirrer, conductivity sensor, heater *etc.*
Kusne et al. [200]	Automated phase mapping and property optimisation for accelerating materials discovery with high-throughput X-ray diffraction	Ge–Sb–Te ternary system	GUI (details not provided)	CAMEO: Matlab-based	Preloaded database (ICSD and AFLOW.org)	Bayesian-based active learning method	Not implemented yet
Burger et al. [35]	Photocatalysts material discovery for hydrogen production from water	Mobile robotic chemist	Java-based GUI	Java-based	CSV file	Bayesian optimiser [169]	KUKA mobile robot to conduct experiments workflow in the lab, *e.g.*, gas chromatograph measurements, solid dispensing, *etc.*
Tran and Ulissi [361]	Computational screening for electrocatalysts discovery of CO_2 reduction and H_2 evolution	Electrochemical reduction	N/A	Python code	MongoDB database	ML method in TPOT package [264]	**Computational:** High-throughput DFT by VASP using ASE on HPC
Christensen et al. [53]	Autonomous process optimisation of a palladium-catalysed stereoselective Suzuki-Miyaura coupling	Batch reactor	N/A	ChemOS: Python-based	Database	Phoenics and Gryffin within ChemOS	Chemspeed SWING robotic system, Agilent 1100
Gao et al. [116]	Computational model development for optimal reaction condition recommendation of organic synthesis reactions	Single-product and single-step reactions	N/A	Python code	Reaxys database	Nerual Network based condiction prediction tool	**Computational:** single NVIDIA GeForce GTX 1080 GPU

Table A.1 (Continued)

Reference	Application	System	Receptionist	Coordinator	Librarian	Planner	Executor
Rosen et al. [305]	Accelerating chemical space exploration of metal–organic frameworks with quantum-chemical calculations and machine learning	Crystalline solids	N/A	Python code	Database	Crystal graph convolutional neural network (CGCNN)	**Computational:** high-throughput DFT by VASP using ASE on HPC
Taylor et al. [353]	Automated determination of reaction models and kinetic parameters	Flow reactor	Matlab-based GUI	Matlab code	Reaction model database in Matlab	MILP optimising reaction kinetics	Tubular reaction vessel built in-house, HPLC pumps, auto-sampler, and HPLC
Waldron et al. [376]	Rapid kinetic model identification	Flow reactor	LabVIEW-based GUI	Python code	CSV file	MBDoE algorithm	Flow reactor, pumps, sampler-dilutor, and HPLC

A.2 Data flow and transfer protocols

Table A.2 Data flow and communication protocols between functional components of the selected state-of-the-art studies in SDLs. The workflow indicates the data flow exchanged within the platform that managed by the coordinator. EP: executor (physical). EC: executor (computational). MS: mass spectrometer. IR: infrared spectroscopy. NMD-M3: Nanotechnology Materials Data Mining, Modeling & Management. HPC: high-performance computing. CRF: chemical recipe file. XDL: chemical description language. VASP: Vienna *ab initio* Simulation Package. ASE: Atomic Simulation Environment.

Reference	Receptionist - Coordinator	Coordinator - Librarian	Coordinator - Planner	Coordinator - Executor	Inner Executor (physical)	Inner Executor (computational)	Workflow (R C L P EP EC)
Fitzpatrick et al. [105]	TCP/IP	MySQL database query (TCP/IP)	In-memory cache	TCP-IP: MS readings transmitted by Arduino-based analogue to serial converter (RS232 serial communication to Ethernet); webcam liquid level position computed by Raspberry Pi in JSON format	Arduino and Raspberry Pi worked as interface for controlling the equipment	N/A	C-[R-P-L-EP-P]

Table A.2 (Continued)

Reference	Receptionist - Coordinator	Coordinator - Librarian	Coordinator - Planner	Coordinator - Executor	Inner Executor (physical)	Inner Executor (computational)	Workflow (R C L P EP EC)
Ingham et al. [172]	HTTP	TCP/IP	Python variables	(1) USB root hub: two webcams; (2) TCP/IP: Uniqsis, Knauer machines and Vapourtec were connected via Brainboxes ES-701 and ES-257 ethernet-to-serial adapters	Raspberry Pi worked as interface for controlling the equipment	N/A	C-[R-P-L-EP-P]
Fitzpatrick et al. [106]	HTTPS	Database query (TCP/IP)	In-memory cache	TCP/IP (with RS232 serial to Ethernet adaptor): equipment was placed within a VLAN that connected to LeyVM server via SSH tunnel	Raspberry Pi worked as interface for controlling the equipment	N/A	C-[R-P-EP-L-P]
Nikolaev et al. [259]	Handled by NMD-M3 software	Handled by NMD-M3 software	Handled by NMD-M3 software	Handled by NMD-M3 software	Through in house built software in C#/.NET	N/A	C-[R-P-EP-L-P]
Wigley et al. [381]	N/A	File transfer	Python variables	File transfer: MAT and textfile	Not specified	N/A	C-[L-P-EP-L]
Greenaway et al. [133]	Within Chemspeed software	File transfer	SSH to HPC	Set the input variables through Chemspeed software	Controlled through Chemspeed software	N/A	C-[EC-L-R-P-EP]
Bédard et al. [17]	Matlab variables	Matlab varialbes	Matlab variable	LabVIEW: through national instrument, serial modem cable, USB cable	Serial command through LabVIEW	N/A	R-P-EP-L-P
Caramelli et al. [43]	N/A	Plaintext as Python variables	Python variables	Python variables handled by gpio and opencv Python libraries	Pumps and webcam are interfaced via pcDuino3 board	N/A	C-[L-P-EP-L]
Skilton et al. [335]	Passed within GLC Solutions	Passed within GLC Solutions	Passed within GLC Solutions	Remote computer control through GLC Solutions	Handled by GLC Solutions	N/A	C-[L-P-E-L]

Table A.2 (Continued)

Reference	Receptionist - Coordinator	Coordinator - Librarian	Coordinator - Planner	Coordinator - Executor	Inner Executor (physical)	Inner Executor (computational)	Workflow (R C L P EP EC)
Coley et al. [59]	CRF file transfer by human researcher	MongoDB database query	Planner generates CRF file to be modified by human researcher	Human researcher pass the CRF file to robotic execution GUI	Universal process bays provide sealing and alignment mechanisms for the fluidic, electrical, and pneumatic process connections	SLURM scheduling software	C-[R(of EC)-L-P(resulted from EC)-R(of EP)-EP]
Roch et al. [303]	JSON, parsed by Python	Database query	Python array	Python pickle object	Raspberry Pi as controller of pumping system, communicated via SCP with the executor codes; Dropbox for synchronising the characterisation equipment	N/A	C-[R-L-P-EP-L]
Montoya et al. [241]	N/A	Python variables	Python variables	Python variables	N/A	AWS Batch API	C-[L-P-EC-L]
Li et al. [209]	TLS encrypted file transfer	SQLAlchemy database query	Python variables (within MAOSIC)	JSON-RPC	Interfaced via high-level and low-level instructions based on JSON-RPC2.0 protocol	N/A	C-[R-L-P-EP-L]
Xue et al. [390]	N/A	Not specified	Python variables	File transfer for DFT	Synthesis performed manually	Handled by Quantum ESPRESSO package	C-[L-P-EC-L]
Cao et al. [41]	N/A	File transfer	File transfer	File transfer	File transfer	N/A	C-[L-P-EP-L]
Jeraal et al. [178]	In-memory cache of Matlab variables	File transfer	Matlab variables	File transfer: CSV file	Interfaced via FlowCommander, a software provided by Vapourtec	N/A	C-[R-L-P-EP-L]
King et al. [190]	N/A	Database query	Not specified	File transfer: LABORS (OWL-DL format)	Closed-source software from Caliper Life Sciences	N/A	C-[L-P-EP-L]

Table A.2 (Continued)

Reference	Receptionist - Coordinator	Coordinator - Librarian	Coordinator - Planner	Coordinator - Executor	Inner Executor (physical)	Inner Executor (computational)	Workflow (R C L P EP EC)
King et al. [189]	N/A	Database query	Not specified	Prolog commands through TCP/IP	Robot operations controlled by tool command language translated from Prolog commands	N/A	C-[L-P-EP-L]
Ingham et al. [171]	HTTP	TCP/IP	Python variables	(1) USB root hub: two webcams; (2) TCP/IP: Uniqsis, Knauer machines and Vapourtec were connected via Brainboxes ES-701 and ES-257 ethernet-to-serial adapters	Raspberry Pi worked as interface for controlling the equipments	N/A	C-[L-P-EP-L]
Schweidtmann et al. [323]	N/A	File transfer	File transfer: CSV file	File transfer: CSV file	Interfaced via FlowCommander, a software developed by Vapourtec	N/A	C-[EP-L-P-EP-L]
HamediRad et al. [142]	N/A	File transfer	Python variables	File transfer: CSV file (iScheduler code)	Managed by iScheduler scheduling software on iBioFAB platform	N/A	C-[L-P-EP-L]
MacLeod et al. [221]	Python variables	Database query	Python variables	Python variables	Driven by North Robotics C9 controller, which also provides auxiliary controls for third-party instruments and components used by the robot	N/A	C-[R-L-P-EP-L]
Segler et al. [325]	N/A	Python variables	Python variables	Python variables	N/A	Theano-backend Keras	C-[L-P-EC]
Steiner et al. [347]	N/A	N/A	XDL (XML-based file) for synthesis route	XDL file	Arduino as micro-controller	N/A	C-[P-EP]
Mehr et al. [229]	In-memory cache & XDL	Textfile	XDL file	XDL file	Arduino as micro-controller	N/A	C-[R-L-R-EP]

Table A.2 (Continued)

Reference	Receptionist - Coordinator	Coordinator - Librarian	Coordinator - Planner	Coordinator - Executor	Inner Executor (physical)	Inner Executor (computational)	Workflow (R C L P EP EC)
Kusne et al. [200]	Not specified	File transfer: MAT file	Matlab variables	Programmatically generated script via SPEC for the SLAC high-throughput system or a GADDS script for the Bruker system	Not specified	N/A	C-[L-P-EP-L]
Burger et al. [35]	N/A	CSV read/write	File transfer	Various communication protocols (TCP/IP over WIFI/LAN; RS-232)	Simultaneous localisation and mapping (SLAM) was used for robot allocation; Arduino designed as micro-controller	N/A	C-[L-P-EP-L]
Tran and Ulissi [361]	N/A	Database query	Python variables	Atom object converted from JSON	N/A	Managed by Luigi and Fireworks	C-[L-P-EC-L]
Christensen et al. [53]	N/A	Database query	Python variables	File transfer (done by a Python script): CSV, textfile & Python pickle object	Chemspeed AutoSuite acts as the control interface	N/A	C-[R-L-P-EP-L]
Gao et al. [116]	N/A	Reaxys API	Python variables	Python variables	N/A	Theano-backend Keras	C-[L-P-EC-L]
Rosen et al. [305]	N/A	Python variables	Python variables	File transfer (ASE argument to VASP)	N/A	Managed by PyMOFScreen	C-[L-P-EC-L]
Taylor et al. [353]	Matlab variables	Matlab variables	Matlab variables	Matlab variables	Matlab variables	N/A	C-[L-P-EP-L]
Waldron et al. [376]	Python variables	File transfer	Python variables	Python variables	LabVIEW acts as the interface	N/A	C-[R-L-P-EP-L]

Appendix B

Ontological representation and software agents

This appendix provides information on the ontologies and software agents involved in this work. We first list the namespaces of the ontologies and then present their description logic representations. Unified Modelling Language (UML) of agents involved in the wet-lab self-optimisation use cases are also provided.

B.1 Namespaces

The namespaces of ontologies in this thesis are categorised into three sections below:

```
## Section 1: common namespaces for general purposes
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix time: <http://www.w3.org/2006/time#> .
@prefix om: <http://www.ontology-of-units-of-measure.org/resource/om-2/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix saref: <https://saref.etsi.org/core/> .
@prefix yago: <http://dbpedia.org/class/yago/> .
@prefix dbo: <https://dbpedia.org/ontology/> .
@prefix dbr: <https://dbpedia.org/resource/> .
## Section 2: OntoCAPE namespaces
@prefix OntoCAPE_Behavior: <http://www.theworldavatar.com/ontology/ontocape/
    chemical_process_system/CPS_behavior/behavior.owl#> .
@prefix OntoCAPE_Material: <http://www.theworldavatar.com/ontology/ontocape/
    material/material.owl#> .
@prefix OntoCAPE_Phase_System: <http://www.theworldavatar.com/ontology/ontocape/
    material/phase_system/phase_system.owl#> .
@prefix OntoCAPE_Reaction_Mechanism: <http://www.theworldavatar.com/ontology/
    ontocape/material/substance/reaction_mechanism.owl#> .
@prefix OntoCAPE_Substance: <http://www.theworldavatar.com/ontology/ontocape/
    material/substance/substance.owl#> .
@prefix OntoCAPE_System: <http://www.theworldavatar.com/ontology/ontocape/
    upper_level/system.owl#> .
## Section 3: The World Avatar namespaces
@prefix OntoSpecies: <http://www.theworldavatar.com/ontology/ontospecies/
    OntoSpecies.owl#> .
@prefix OntoKin: <http://www.theworldavatar.com/ontology/ontokin/OntoKin.owl#> .
@prefix OntoReaction: <https://www.theworldavatar.com/kg/ontoreaction/> .
@prefix OntoDoE: <https://www.theworldavatar.com/kg/ontodoe/> .
@prefix OntoLab: <https://www.theworldavatar.com/kg/ontolab/> .
@prefix OntoVapourtec: <https://www.theworldavatar.com/kg/ontovapourtec/> .
@prefix OntoHPLC: <https://www.theworldavatar.com/kg/ontohplc/> .
@prefix OntoDerivation: <https://www.theworldavatar.com/kg/ontoderivation/> .
@prefix OntoGoal: <https://www.theworldavatar.com/kg/ontogoal/> .
@prefix OntoAgent: <http://www.theworldavatar.com/ontology/ontoagent/MSM.owl#> .
```

Table B.1 Short names and their corresponding namespaces in OntoCAPE.

Short name	**Corresponding namespace**
MaterialAmount	OntoCAPE_Behavior
Material	OntoCAPE_Material
SinglePhase	OntoCAPE_Phase_System
PhysicalContext	OntoCAPE_Phase_System
StateOfAggregation	OntoCAPE_Phase_System
PhaseComponent	OntoCAPE_Phase_System
ChemicalSpecies	OntoCAPE_Substance
Composition	OntoCAPE_Phase_System
PhaseComponentConcentration	OntoCAPE_Phase_System
Volume-BasedConcentration	OntoCAPE_Phase_System
refersToMaterial	OntoCAPE_Behavior
thermodynamicBehavior	OntoCAPE_Material
representsThermodynamicBehaviorOf	OntoCAPE_Material
hasStateOfAggregation	OntoCAPE_Phase_System
has_physical_context	OntoCAPE_Phase_System
isComposedOfSubsystem	OntoCAPE_System
representsOccurenceOf	OntoCAPE_Phase_System
hasProperty	OntoCAPE_System
has_composition	OntoCAPE_Phase_System
comprisesDirectly	OntoCAPE_System
liquid	OntoCAPE_Phase_System
gaseous	OntoCAPE_Phase_System
solid	OntoCAPE_Phase_System

B.2 Description logic representation

The description logic representations of the ontologies developed in this work are provided below.

OntoDerivation
Classes:
Error $\sqsubseteq$ Status
Finished $\sqsubseteq$ Status
InProgress $\sqsubseteq$ Status
Requested $\sqsubseteq$ Status

Object properties:
$\exists$ belongsTo.$\top \sqsubseteq \top$
$\exists$ hasNewDerivedIRI.$\top \sqsubseteq$ Finished
$\exists$ hasStatus.$\top \sqsubseteq$ DerivationAsyn
$\exists$ isDerivedFrom.$\top \sqsubseteq$ (Derivation $\sqcup$ DerivationAsyn $\sqcup$ DerivationWithTimeSeries)
$\exists$ isDerivedUsing.$\top \sqsubseteq$ (Derivation $\sqcup$ DerivationAsyn $\sqcup$ DerivationWithTimeSeries)
$\top \sqsubseteq \forall$ belongsTo.(Derivation $\sqcup$ DerivationAsyn $\sqcup$ DerivationWithTimeSeries)
$\top \sqsubseteq \forall$ hasNewDerivedIRI.$\top$
$\top \sqsubseteq \forall$ hasStatus.Status
$\top \sqsubseteq \forall$ isDerivedFrom.$\top$
$\top \sqsubseteq \forall$ isDerivedUsing.OntoAgent:Service

Data properties:
$\exists$ retrievedInputsAt.$\top \sqsubseteq$ DerivationAsyn
$\exists$ uuidLock.$\top \sqsubseteq$ DerivationAsyn
$\top \sqsubseteq \forall$ retrievedInputsAt.xsd:decimal
$\top \sqsubseteq \forall$ uuidLock.xsd:string

OntoReaction
Classes:
OntoReaction:Base $\sqsubseteq$ OntoKin:Species
OntoReaction:Catalyst $\sqsubseteq$ OntoKin:Species
OntoReaction:Chemical $\sqsubseteq$ OntoCAPE_Material:Material
OntoReaction:ChemicalReaction $\sqsubseteq$ OntoCAPE_Reaction_Mechanism:ChemicalReaction

OntoReaction:Conversion ⊑ om:AmountOfSubstanceFraction
OntoReaction:Conversion ⊑ OntoReaction:PerformanceIndicator
OntoReaction:Conversion ⊑ = 1 OntoReaction:yieldLimitingSpecies.OntoSpecies:Species
OntoReaction:EcoScore ⊑ om:QuantityOfDimensionOne
OntoReaction:EcoScore ⊑ OntoReaction:PerformanceIndicator
OntoReaction:EnvironmentalFactor ⊑ om:MassFraction
OntoReaction:EnvironmentalFactor ⊑ OntoReaction:PerformanceIndicator
OntoReaction:Impurity ⊑ OntoKin:Product
OntoReaction:InputChemical ⊑ OntoReaction:Chemical
OntoReaction:OutputChemical ⊑ OntoReaction:Chemical
OntoReaction:ReactionExperiment ⊑ = 1 OntoReaction:hasResTime.OntoReaction:ResidenceTime
OntoReaction:ReactionExperiment ⊑ = 1 OntoReaction:hasRxnPressure.OntoReaction:ReactionPressure
OntoReaction:ReactionExperiment ⊑ = 1 OntoReaction:hasRxnScale.OntoReaction:ReactionScale
OntoReaction:ReactionExperiment ⊑ = 1 OntoReaction:hasRxnTemperature.OntoReaction:ReactionTemperature
OntoReaction:ReactionExperiment ⊑ = 1 OntoReaction:isOccurenceOf.OntoReaction:ChemicalReaction
OntoReaction:ReactionPressure ⊑ om:Pressure
OntoReaction:ReactionPressure ⊑ OntoReaction:ReactionCondition
OntoReaction:ReactionScale ⊑ om:Volume
OntoReaction:ReactionScale ⊑ OntoReaction:ReactionCondition
OntoReaction:ReactionScale ⊑ = 1 OntoReaction:indicatesUsageOf.OntoReaction:InputChemical
OntoReaction:ReactionTemperature ⊑ om:CelsiusTemperature
OntoReaction:ReactionTemperature ⊑ OntoReaction:ReactionCondition
OntoReaction:ReactionVariation ⊑ OntoReaction:ReactionExperiment
OntoReaction:ReactionVariation ⊑ = 1 OntoReaction:isVariationOf.OntoReaction:ReactionExperiment
OntoReaction:ResidenceTime ⊑ om:Duration
OntoReaction:ResidenceTime ⊑ OntoReaction:ReactionCondition
OntoReaction:RunMaterialCost ⊑ om:SpecificAmountOfMoney
OntoReaction:RunMaterialCost ⊑ OntoReaction:PerformanceIndicator
OntoReaction:RunMaterialCostPerKilogramProduct ⊑ om:SpecificAmountOfMoney

OntoReaction:RunMaterialCostPerKilogramProduct ⊑ OntoReaction:PerformanceIndicator
OntoReaction:Solvent ⊑ OntoKin:Species
OntoReaction:SpaceTimeYield ⊑ om:Quantity
OntoReaction:SpaceTimeYield ⊑ OntoReaction:PerformanceIndicator
OntoReaction:StoichiometryRatio ⊑ om:VolumeFraction
OntoReaction:StoichiometryRatio ⊑ OntoReaction:ReactionCondition
OntoReaction:StoichiometryRatio ⊑ = 1 OntoReaction:indicatesMultiplicityOf.OntoReaction:InputChemical
OntoReaction:TargetProduct ⊑ OntoKin:Product
OntoReaction:Yield ⊑ om:AmountOfSubstanceFraction
OntoReaction:Yield ⊑ OntoReaction:PerformanceIndicator
OntoReaction:Yield ⊑ = 1 OntoReaction:yieldLimitingSpecies.OntoSpecies:Species

Object properties:
OntoReaction:hasCatalyst ⊑ OntoCAPE_Reaction_Mechanism:hasCatalyst
OntoReaction:hasConversion ⊑ OntoReaction:hasPerformanceIndicator
OntoReaction:hasEcoScore ⊑ OntoReaction:hasPerformanceIndicator
OntoReaction:hasEnvironmentalFactor ⊑ OntoReaction:hasPerformanceIndicator
OntoReaction:hasResTime ⊑ OntoReaction:hasReactionCondition
OntoReaction:hasRunMaterialCost ⊑ OntoReaction:hasPerformanceIndicator
OntoReaction:hasRunMaterialCostPerKilogramProduct ⊑ OntoReaction:hasPerformanceIndicator
OntoReaction:hasRxnPressure ⊑ OntoReaction:hasReactionCondition
OntoReaction:hasRxnScale ⊑ OntoReaction:hasReactionCondition
OntoReaction:hasRxnTemperature ⊑ OntoReaction:hasReactionCondition
OntoReaction:hasSpaceTimeYield ⊑ OntoReaction:hasPerformanceIndicator
OntoReaction:hasStoichiometryRatio ⊑ OntoReaction:hasReactionCondition
OntoReaction:hasYield ⊑ OntoReaction:hasPerformanceIndicator
∃ OntoReaction:hasBase.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:hasCatalyst.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:hasConversion.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasEcoScore.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasEnvironmentalFactor.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasInputChemical.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasOutputChemical.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasPerformanceIndicator.⊤ ⊑ OntoReaction:ReactionExperiment

∃ OntoReaction:hasReactionCondition.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasResTime.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasRunMaterialCost.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasRunMaterialCostPerKilogramProduct.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasRxnPressure.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasRxnScale.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasRxnTemperature.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasRxnType.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:hasSolvent.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:hasSpaceTimeYield.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasStoichiometryRatio.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:hasYield.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:indicatesMultiplicityOf.⊤ ⊑ OntoReaction:StoichiometryRatio
∃ OntoReaction:indicatesUsageOf.⊤ ⊑ OntoReaction:ReactionScale
∃ OntoReaction:isAssignedTo.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:isOccurenceOf.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoReaction:isVariationOf.⊤ ⊑ OntoReaction:ReactionVariation
∃ OntoReaction:yieldLimitingSpecies.⊤ ⊑ (OntoReaction:Conversion ⊔ OntoReaction:Yield)
⊤ ⊑ ∀ OntoReaction:hasBase.OntoReaction:Base
⊤ ⊑ ∀ OntoReaction:hasCatalyst.OntoReaction:Catalyst
⊤ ⊑ ∀ OntoReaction:hasConversion.OntoReaction:Conversion
⊤ ⊑ ∀ OntoReaction:hasEcoScore.OntoReaction:EcoScore
⊤ ⊑ ∀ OntoReaction:hasEnvironmentalFactor.OntoReaction:EnvironmentalFactor
⊤ ⊑ ∀ OntoReaction:hasInputChemical.OntoReaction:InputChemical
⊤ ⊑ ∀ OntoReaction:hasOutputChemical.OntoReaction:OutputChemical
⊤ ⊑ ∀ OntoReaction:hasPerformanceIndicator.OntoReaction:PerformanceIndicator
⊤ ⊑ ∀ OntoReaction:hasReactionCondition.OntoReaction:ReactionCondition
⊤ ⊑ ∀ OntoReaction:hasResTime.OntoReaction:ResidenceTime
⊤ ⊑ ∀ OntoReaction:hasRunMaterialCost.OntoReaction:RunMaterialCost
⊤ ⊑ ∀ OntoReaction:hasRunMaterialCostPerKilogramProduct.OntoReaction:RunMaterialCostPerKilogramProduct
⊤ ⊑ ∀ OntoReaction:hasRxnPressure.OntoReaction:ReactionPressure
⊤ ⊑ ∀ OntoReaction:hasRxnScale.OntoReaction:ReactionScale
⊤ ⊑ ∀ OntoReaction:hasRxnTemperature.OntoReaction:ReactionTemperature

⊤ ⊑ ∀ OntoReaction:hasRxnType.RXNO:MolecularProcess
⊤ ⊑ ∀ OntoReaction:hasSolvent.OntoReaction:Solvent
⊤ ⊑ ∀ OntoReaction:hasSpaceTimeYield.OntoReaction:SpaceTimeYield
⊤ ⊑ ∀ OntoReaction:hasStoichiometryRatio.OntoReaction:StoichiometryRatio
⊤ ⊑ ∀ OntoReaction:hasYield.OntoReaction:Yield
⊤ ⊑ ∀ OntoReaction:indicatesMultiplicityOf.OntoReaction:InputChemical
⊤ ⊑ ∀ OntoReaction:indicatesUsageOf.OntoReaction:InputChemical
⊤ ⊑ ∀ OntoReaction:isAssignedTo.OntoVapourtec:VapourtecR4Reactor
⊤ ⊑ ∀ OntoReaction:isOccurenceOf.OntoReaction:ChemicalReaction
⊤ ⊑ ∀ OntoReaction:isVariationOf.OntoReaction:ReactionExperiment
⊤ ⊑ ∀ OntoReaction:yieldLimitingSpecies.OntoSpecies:Species

Data properties:
∃ OntoReaction:cdXML.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:hasRDFILE.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:hasRInChI.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:ordID.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:rxnCXSMILES.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoReaction:rxnSMILES.⊤ ⊑ OntoReaction:ChemicalReaction
⊤ ⊑ ∀ OntoReaction:cdXML.xsd:string
⊤ ⊑ ∀ OntoReaction:hasRDFILE.xsd:string
⊤ ⊑ ∀ OntoReaction:hasRInChI.xsd:string
⊤ ⊑ ∀ OntoReaction:ordID.xsd:string
⊤ ⊑ ∀ OntoReaction:rxnCXSMILES.xsd:string
⊤ ⊑ ∀ OntoReaction:rxnSMILES.xsd:string

OntoDoE
Classes:
OntoDoE:CategoricalVariable ⊑ OntoDoE:DesignVariable
OntoDoE:CategoricalVariable ⊑ = 1 OntoDoE:refersToQuantity.om:Quantity
OntoDoE:Center ⊑ OntoDoE:Criterion
OntoDoE:CenterMaximum ⊑ OntoDoE:Criterion
OntoDoE:ContinuousVariable ⊑ OntoDoE:DesignVariable
OntoDoE:ContinuousVariable ⊑ = 1 OntoDoE:refersToQuantity.om:Quantity
OntoDoE:Correlation ⊑ OntoDoE:Criterion

OntoDoE:DesignOfExperiment ⊑ ≥ 1 OntoDoE:hasSystemResponse.OntoDoE:SystemResponse
OntoDoE:DesignOfExperiment ⊑ = 1 OntoDoE:hasDomain.OntoDoE:Domain
OntoDoE:DesignOfExperiment ⊑ = 1 OntoDoE:usesStrategy.OntoDoE:Strategy
OntoDoE:DesignOfExperiment ⊑ = 1 OntoDoE:utilisesHistoricalData.OntoDoE:HistoricalData
OntoDoE:Domain ⊑ ≥ 1 OntoDoE:hasDesignVariable.OntoDoE:DesignVariable
OntoDoE:FixedParameter ⊑ = 1 OntoDoE:refersToQuantity.om:Quantity
OntoDoE:LHS ⊑ OntoDoE:Strategy
OntoDoE:Maximum ⊑ OntoDoE:Criterion
OntoDoE:NewSTBO ⊑ OntoDoE:Strategy
OntoDoE:SystemResponse ⊑ = 1 OntoDoE:refersToQuantity.om:Quantity
OntoDoE:TSEMO ⊑ OntoDoE:Strategy

Object properties:
∃ OntoDoE:designsChemicalReaction.⊤ ⊑ OntoDoE:DesignOfExperiment
∃ OntoDoE:hasDesignVariable.⊤ ⊑ OntoDoE:Domain
∃ OntoDoE:hasDoETemplate.⊤ ⊑ OntoReaction:ChemicalReaction
∃ OntoDoE:hasDomain.⊤ ⊑ OntoDoE:DesignOfExperiment
∃ OntoDoE:hasFixedParameter.⊤ ⊑ OntoDoE:Domain
∃ OntoDoE:hasSystemResponse.⊤ ⊑ OntoDoE:DesignOfExperiment
∃ OntoDoE:proposesNewExperiment.⊤ ⊑ OntoDoE:DesignOfExperiment
∃ OntoDoE:refersToExperiment.⊤ ⊑ OntoDoE:HistoricalData
∃ OntoDoE:refersToQuantity.⊤ ⊑ (OntoDoE:CategoricalVariable ⊔ OntoDoE:ContinuousVariable ⊔ OntoDoE:FixedParameter ⊔ OntoDoE:SystemResponse)
∃ OntoDoE:usesStrategy.⊤ ⊑ OntoDoE:DesignOfExperiment
∃ OntoDoE:utilisesHistoricalData.⊤ ⊑ OntoDoE:DesignOfExperiment
⊤ ⊑ ∀ OntoDoE:designsChemicalReaction.OntoReaction:ChemicalReaction
⊤ ⊑ ∀ OntoDoE:hasDesignVariable.OntoDoE:DesignVariable
⊤ ⊑ ∀ OntoDoE:hasDoETemplate.OntoDoE:DesignOfExperiment
⊤ ⊑ ∀ OntoDoE:hasDomain.OntoDoE:Domain
⊤ ⊑ ∀ OntoDoE:hasFixedParameter.OntoDoE:FixedParameter
⊤ ⊑ ∀ OntoDoE:hasSystemResponse.OntoDoE:SystemResponse
⊤ ⊑ ∀ OntoDoE:proposesNewExperiment.OntoReaction:ReactionExperiment
⊤ ⊑ ∀ OntoDoE:refersToExperiment.OntoReaction:ReactionExperiment
⊤ ⊑ ∀ OntoDoE:refersToQuantity.om:Quantity

$\top \sqsubseteq \forall$ OntoDoE:usesStrategy.OntoDoE:Strategy
$\top \sqsubseteq \forall$ OntoDoE:utilisesHistoricalData.OntoDoE:HistoricalData

Data properties:
$\exists$ OntoDoE:hasLevel.$\top \sqsubseteq$ OntoDoE:CategoricalVariable
$\exists$ OntoDoE:lowerLimit.$\top \sqsubseteq$ OntoDoE:ContinuousVariable
$\exists$ OntoDoE:maximise.$\top \sqsubseteq$ OntoDoE:SystemResponse
$\exists$ OntoDoE:nGenerations.$\top \sqsubseteq$ OntoDoE:TSEMO
$\exists$ OntoDoE:nRetries.$\top \sqsubseteq$ OntoDoE:TSEMO
$\exists$ OntoDoE:nSpectralPoints.$\top \sqsubseteq$ OntoDoE:TSEMO
$\exists$ OntoDoE:numOfNewExp.$\top \sqsubseteq$ OntoDoE:HistoricalData
$\exists$ OntoDoE:populationSize.$\top \sqsubseteq$ OntoDoE:TSEMO
$\exists$ OntoDoE:positionalID.$\top \sqsubseteq$ (OntoDoE:ContinuousVariable $\sqcup$ OntoDoE:FixedParameter $\sqcup$ OntoDoE:SystemResponse)
$\exists$ OntoDoE:seed.$\top \sqsubseteq$ OntoDoE:LHS
$\exists$ OntoDoE:upperLimit.$\top \sqsubseteq$ OntoDoE:ContinuousVariable
$\top \sqsubseteq \forall$ OntoDoE:hasLevel.xsd:string
$\top \sqsubseteq \forall$ OntoDoE:lowerLimit.xsd:float
$\top \sqsubseteq \forall$ OntoDoE:maximise.xsd:boolean
$\top \sqsubseteq \forall$ OntoDoE:nGenerations.xsd:int
$\top \sqsubseteq \forall$ OntoDoE:nRetries.xsd:int
$\top \sqsubseteq \forall$ OntoDoE:nSpectralPoints.xsd:int
$\top \sqsubseteq \forall$ OntoDoE:numOfNewExp.xsd:int
$\top \sqsubseteq \forall$ OntoDoE:populationSize.xsd:int
$\top \sqsubseteq \forall$ OntoDoE:positionalID.xsd:string
$\top \sqsubseteq \forall$ OntoDoE:seed.xsd:int
$\top \sqsubseteq \forall$ OntoDoE:upperLimit.xsd:float

OntoLab
Classes:
OntoLab:ChemicalAmount $\sqsubseteq$ OntoCAPE_Behavior:MaterialAmount
OntoLab:ChemicalContainer $\sqsubseteq = 1$ OntoLab:hasFillLevel.om:Volume
OntoLab:ChemicalContainer $\sqsubseteq = 1$ OntoLab:hasMaxLevel.om:Volume
OntoLab:ChemicalContainer $\sqsubseteq = 1$ OntoLab:hasWarningLevel.om:Volume
OntoLab:ChemicalContainer $\sqsubseteq \leq 1$ OntoLab:isFilledWith.OntoLab:ChemicalAmount

OntoLab:Dried $\sqsubseteq$ OntoLab:PreparationMethod
OntoLab:DurationSetting $\sqsubseteq$ OntoLab:ParameterSetting
OntoLab:ExternalBattery $\sqsubseteq$ OntoLab:PowerSupply
OntoLab:ExternalDC $\sqsubseteq$ OntoLab:PowerSupply
OntoLab:FlowRateSetting $\sqsubseteq$ OntoLab:ParameterSetting
OntoLab:LabEquipment $\sqsubseteq$ saref:Device
OntoLab:LabEquipment $\sqsubseteq \geq 1$ OntoLab:hasPowerSupply.OntoLab:PowerSupply
OntoLab:LabEquipment $\sqsubseteq \leq 1$ OntoLab:hasHeight.om:Height
OntoLab:LabEquipment $\sqsubseteq \leq 1$ OntoLab:hasLength.om:Length
OntoLab:LabEquipment $\sqsubseteq \leq 1$ OntoLab:hasPrice.om:AmountOfMoney
OntoLab:LabEquipment $\sqsubseteq \leq 1$ OntoLab:hasWeight.om:BodyMass
OntoLab:LabEquipment $\sqsubseteq \leq 1$ OntoLab:hasWidth.om:Width
OntoLab:LabEquipment $\sqsubseteq \leq 1$ OntoLab:isManagedBy.OntoAgent:Service
OntoLab:LithiumBattery $\sqsubseteq$ OntoLab:PowerSupply
OntoLab:NiMHRechargeableBattery $\sqsubseteq$ OntoLab:PowerSupply
OntoLab:ParameterSetting $\sqsubseteq \leq 1$ OntoLab:hasQuantity.om:Quantity
OntoLab:ReagentBottle $\sqsubseteq$ OntoLab:ChemicalContainer
OntoLab:Repurified $\sqsubseteq$ OntoLab:PreparationMethod
OntoLab:SolarPowerPack $\sqsubseteq$ OntoLab:PowerSupply
OntoLab:Sparged $\sqsubseteq$ OntoLab:PreparationMethod
OntoLab:SynthesisedInHouse $\sqsubseteq$ OntoLab:PreparationMethod
OntoLab:TemperatureSetting $\sqsubseteq$ OntoLab:ParameterSetting
OntoLab:UsedAsReceived $\sqsubseteq$ OntoLab:PreparationMethod
OntoLab:Vial $\sqsubseteq$ OntoLab:ChemicalContainer
OntoLab:VolumeSetting $\sqsubseteq$ OntoLab:ParameterSetting
OntoLab:WasteBottle $\sqsubseteq$ OntoLab:ChemicalContainer

Object properties:
$\exists$ OntoLab:contains.$\top \sqsubseteq$ OntoLab:Laboratory
$\exists$ OntoLab:hasFillLevel.$\top \sqsubseteq$ OntoLab:ChemicalContainer
$\exists$ OntoLab:hasHeight.$\top \sqsubseteq$ OntoLab:LabEquipment
$\exists$ OntoLab:hasLength.$\top \sqsubseteq$ OntoLab:LabEquipment
$\exists$ OntoLab:hasMaxLevel.$\top \sqsubseteq$ OntoLab:ChemicalContainer
$\exists$ OntoLab:hasPowerSupply.$\top \sqsubseteq$ OntoLab:LabEquipment
$\exists$ OntoLab:hasPrice.$\top \sqsubseteq$ OntoLab:LabEquipment
$\exists$ OntoLab:hasQuantity.$\top \sqsubseteq$ OntoLab:ParameterSetting

∃ OntoLab:hasSetting.⊤ ⊑ OntoLab:EquipmentSettings
∃ OntoLab:hasWarningLevel.⊤ ⊑ OntoLab:ChemicalContainer
∃ OntoLab:hasWeight.⊤ ⊑ OntoLab:LabEquipment
∃ OntoLab:hasWidth.⊤ ⊑ OntoLab:LabEquipment
∃ OntoLab:isFilledWith.⊤ ⊑ OntoLab:ChemicalContainer
∃ OntoLab:isManagedBy.⊤ ⊑ OntoLab:LabEquipment
∃ OntoLab:isPreparedBy.⊤ ⊑ OntoLab:ChemicalAmount
∃ OntoLab:specifies.⊤ ⊑ OntoLab:EquipmentSettings
∃ OntoLab:wasGeneratedFor.⊤ ⊑ OntoLab:EquipmentSettings
⊤ ⊑ ∀ OntoLab:contains.OntoLab:LabEquipment
⊤ ⊑ ∀ OntoLab:hasFillLevel.om:Volume
⊤ ⊑ ∀ OntoLab:hasHeight.om:Height
⊤ ⊑ ∀ OntoLab:hasLength.om:Length
⊤ ⊑ ∀ OntoLab:hasMaxLevel.om:Volume
⊤ ⊑ ∀ OntoLab:hasPowerSupply.OntoLab:PowerSupply
⊤ ⊑ ∀ OntoLab:hasPrice.om:AmountOfMoney
⊤ ⊑ ∀ OntoLab:hasQuantity.om:Quantity
⊤ ⊑ ∀ OntoLab:hasSetting.OntoLab:ParameterSetting
⊤ ⊑ ∀ OntoLab:hasWarningLevel.om:Volume
⊤ ⊑ ∀ OntoLab:hasWeight.om:BodyMass
⊤ ⊑ ∀ OntoLab:hasWidth.om:Width
⊤ ⊑ ∀ OntoLab:isFilledWith.OntoLab:ChemicalAmount
⊤ ⊑ ∀ OntoLab:isManagedBy.OntoAgent:Service
⊤ ⊑ ∀ OntoLab:isPreparedBy.OntoLab:PreparationMethod
⊤ ⊑ ∀ OntoLab:specifies.OntoLab:LabEquipment
⊤ ⊑ ∀ OntoLab:wasGeneratedFor.OntoReaction:ReactionExperiment

Data properties:
∃ OntoLab:containsUnidentifiedComponent.⊤ ⊑ OntoLab:ChemicalAmount
∃ OntoLab:stateLastUpdatedAt.⊤ ⊑ saref:State
⊤ ⊑ ∀ OntoLab:containsUnidentifiedComponent.xsd:boolean
⊤ ⊑ ∀ OntoLab:stateLastUpdatedAt.xsd:decimal

OntoVapourtec

Classes:

OntoVapourtec:AutoSampler $\sqsubseteq$ OntoLab:LabEquipment
OntoVapourtec:AutoSamplerCommand $\sqsubseteq$ saref:Command
OntoVapourtec:AutoSamplerFunction $\sqsubseteq$ saref:Function
OntoVapourtec:AutoSamplerSite $\sqsubseteq \leq 1$ OntoVapourtec:holds.OntoLab:Vial
OntoVapourtec:AutoSamplerTask $\sqsubseteq$ saref:Task
OntoVapourtec:CleanReactor $\sqsubseteq$ OntoVapourtec:VapourtecFunction
OntoVapourtec:CleaningReaction $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:ClearReactions $\sqsubseteq$ saref:Command
OntoVapourtec:ConnectToFlowCommander $\sqsubseteq$ saref:Command
OntoVapourtec:Connection $\sqsubseteq$ OntoVapourtec:VapourtecFunction
OntoVapourtec:DryRunState $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:ExpFilePath $\sqsubseteq$ OntoLab:Argument
OntoVapourtec:FaultRecovery $\sqsubseteq$ OntoVapourtec:VapourtecFunction
OntoVapourtec:FaultRecoveryCommand $\sqsubseteq$ saref:Command
OntoVapourtec:Faulty $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:FinalCleaning $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:FlowChemistry $\sqsubseteq$ saref:Task
OntoVapourtec:FlowCommander $\sqsubseteq$ saref:Service
OntoVapourtec:FractionCollector $\sqsubseteq$ OntoVapourtec:CollectionMethod
OntoVapourtec:GetCommand $\sqsubseteq$ saref:Command
OntoVapourtec:GetState $\sqsubseteq$ OntoVapourtec:VapourtecFunction
OntoVapourtec:Idle $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:Inactive $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:Initialising $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:Launch $\sqsubseteq$ OntoVapourtec:VapourtecFunction
OntoVapourtec:LoadExperiment $\sqsubseteq$ saref:Command
OntoVapourtec:Null $\sqsubseteq$ OntoVapourtec:VapourtecState
OntoVapourtec:PumpSettings $\sqsubseteq$ OntoLab:EquipmentSettings
OntoVapourtec:PumpSettings $\sqsubseteq \leq 1$ OntoVapourtec:hasFlowRateSetting.OntoLab:FlowRateSetting
OntoVapourtec:PumpSettings $\sqsubseteq \leq 1$ OntoVapourtec:hasSampleLoopVolumeSetting.OntoVapourtec:SampleLoopVolumeSetting
OntoVapourtec:PumpSettings $\sqsubseteq \leq 1$ OntoVapourtec:hasStoichiometryRatioSetting.OntoVapourtec:StoichiometryRatioSetting
OntoVapourtec:ReactionCompleted $\sqsubseteq$ OntoVapourtec:VapourtecState

OntoVapourtec:ReactorSettings ⊑ OntoLab:EquipmentSettings
OntoVapourtec:ReactorSettings ⊑ ≤ 1 OntoVapourtec:hasReactorTemperatureSetting.OntoVapourtec:ReactorTemperatureSetting
OntoVapourtec:ReactorSettings ⊑ ≤ 1 OntoVapourtec:hasResidenceTimeSetting.OntoVapourtec:ResidenceTimeSetting
OntoVapourtec:ReactorTemperatureSetting ⊑ OntoLab:TemperatureSetting
OntoVapourtec:ResidenceTimeSetting ⊑ OntoLab:DurationSetting
OntoVapourtec:RunReactor ⊑ OntoVapourtec:VapourtecFunction
OntoVapourtec:RunningReaction ⊑ OntoVapourtec:VapourtecState
OntoVapourtec:SampleLoopVolumeSetting ⊑ OntoLab:VolumeSetting
OntoVapourtec:SingleReceptacle ⊑ OntoVapourtec:CollectionMethod
OntoVapourtec:StartFlowCommander ⊑ saref:Command
OntoVapourtec:StoichiometryRatioSetting ⊑ OntoLab:ParameterSetting
OntoVapourtec:VapourtecFunction ⊑ saref:Function
OntoVapourtec:VapourtecR2Pump ⊑ OntoLab:LabEquipment
OntoVapourtec:VapourtecR2PumpCommand ⊑ saref:Command
OntoVapourtec:VapourtecR2PumpFunction ⊑ saref:Function
OntoVapourtec:VapourtecR2PumpTask ⊑ saref:Task
OntoVapourtec:VapourtecR4Reactor ⊑ OntoLab:LabEquipment
OntoVapourtec:VapourtecR4Reactor ⊑ = 1 OntoVapourtec:hasInternalDiameter.om:Diameter
OntoVapourtec:VapourtecR4Reactor ⊑ = 1 OntoVapourtec:hasReactorLength.om:Length
OntoVapourtec:VapourtecR4Reactor ⊑ = 1 OntoVapourtec:hasReactorTemperatureLowerLimit.om:CelsiusTemperature
OntoVapourtec:VapourtecR4Reactor ⊑ = 1 OntoVapourtec:hasReactorTemperatureUpperLimit.om:CelsiusTemperature
OntoVapourtec:VapourtecR4Reactor ⊑ = 1 OntoVapourtec:hasReactorVolume.om:Volume
OntoVapourtec:VapourtecR4ReactorCommand ⊑ saref:Command
OntoVapourtec:VapourtecR4ReactorFunction ⊑ saref:Function
OntoVapourtec:VapourtecR4ReactorTask ⊑ saref:Task
OntoVapourtec:VapourtecRS400 ⊑ OntoLab:LabEquipment
OntoVapourtec:VapourtecRS400 ⊑ = 1 OntoVapourtec:hasCollectionMethod.OntoVapourtec:CollectionMethod
OntoVapourtec:VapourtecRS400 ⊑ = 1 OntoVapourtec:recommendedReactionScale.om:Volume

OntoVapourtec:VapourtecState ⊑ saref:State

Object properties:
OntoVapourtec:hasFlowRateSetting ⊑ OntoLab:hasSetting
OntoVapourtec:hasReactorTemperatureSetting ⊑ OntoLab:hasSetting
OntoVapourtec:hasResidenceTimeSetting ⊑ OntoLab:hasSetting
OntoVapourtec:hasSampleLoopVolumeSetting ⊑ OntoLab:hasSetting
OntoVapourtec:hasStoichiometryRatioSetting ⊑ OntoLab:hasSetting
∃ OntoVapourtec:hasCollectionMethod.⊤ ⊑ OntoVapourtec:VapourtecRS400
∃ OntoVapourtec:hasFlowRateSetting.⊤ ⊑ OntoVapourtec:PumpSettings
∃ OntoVapourtec:hasInternalDiameter.⊤ ⊑ OntoVapourtec:VapourtecR4Reactor
∃ OntoVapourtec:hasReactorLength.⊤ ⊑ OntoVapourtec:VapourtecR4Reactor
∃ OntoVapourtec:hasReactorMaterial.⊤ ⊑ OntoVapourtec:VapourtecR4Reactor
∃ OntoVapourtec:hasReactorTemperatureLowerLimit.⊤ ⊑ OntoVapourtec:VapourtecR4Reactor
∃ OntoVapourtec:hasReactorTemperatureSetting.⊤ ⊑ OntoVapourtec:ReactorSettings
∃ OntoVapourtec:hasReactorTemperatureUpperLimit.⊤ ⊑ OntoVapourtec:VapourtecR4Reactor
∃ OntoVapourtec:hasReactorVolume.⊤ ⊑ OntoVapourtec:VapourtecR4Reactor
∃ OntoVapourtec:hasReagentSource.⊤ ⊑ OntoVapourtec:VapourtecR2Pump
∃ OntoVapourtec:hasResidenceTimeSetting.⊤ ⊑ OntoVapourtec:ReactorSettings
∃ OntoVapourtec:hasSampleLoopVolumeSetting.⊤ ⊑ OntoVapourtec:PumpSettings
∃ OntoVapourtec:hasSite.⊤ ⊑ OntoVapourtec:AutoSampler
∃ OntoVapourtec:hasStoichiometryRatioSetting.⊤ ⊑ OntoVapourtec:PumpSettings
∃ OntoVapourtec:hasVapourtecInputFile.⊤ ⊑ OntoReaction:ReactionExperiment
∃ OntoVapourtec:holds.⊤ ⊑ OntoVapourtec:AutoSamplerSite
∃ OntoVapourtec:pumpsLiquidFrom.⊤ ⊑ OntoVapourtec:PumpSettings
∃ OntoVapourtec:recommendedReactionScale.⊤ ⊑ OntoVapourtec:VapourtecRS400
∃ OntoVapourtec:sampleLoopVolume.⊤ ⊑ OntoVapourtec:AutoSampler
∃ OntoVapourtec:toReceptacle.⊤ ⊑ OntoVapourtec:SingleReceptacle
⊤ ⊑ ∀ OntoVapourtec:hasCollectionMethod.OntoVapourtec:CollectionMethod
⊤ ⊑ ∀ OntoVapourtec:hasFlowRateSetting.OntoLab:FlowRateSetting
⊤ ⊑ ∀ OntoVapourtec:hasInternalDiameter.om:Diameter
⊤ ⊑ ∀ OntoVapourtec:hasReactorLength.om:Length
⊤ ⊑ ∀ OntoVapourtec:hasReactorMaterial.OntoCAPE_Behavior:MaterialAmount
⊤ ⊑ ∀ OntoVapourtec:hasReactorTemperatureLowerLimit.om:CelsiusTemperature

⊤ ⊑ ∀ OntoVapourtec:hasReactorTemperatureSetting.OntoVapourtec:ReactorTemperatureSetting
⊤ ⊑ ∀ OntoVapourtec:hasReactorTemperatureUpperLimit.om:CelsiusTemperature
⊤ ⊑ ∀ OntoVapourtec:hasReactorVolume.om:Volume
⊤ ⊑ ∀ OntoVapourtec:hasReagentSource.OntoLab:ReagentBottle
⊤ ⊑ ∀ OntoVapourtec:hasResidenceTimeSetting.OntoVapourtec:ResidenceTimeSetting
⊤ ⊑ ∀ OntoVapourtec:hasSampleLoopVolumeSetting.OntoVapourtec:SampleLoopVolumeSetting
⊤ ⊑ ∀ OntoVapourtec:hasSite.OntoVapourtec:AutoSamplerSite
⊤ ⊑ ∀ OntoVapourtec:hasStoichiometryRatioSetting.OntoVapourtec:StoichiometryRatioSetting
⊤ ⊑ ∀ OntoVapourtec:hasVapourtecInputFile.OntoVapourtec:VapourtecInputFile
⊤ ⊑ ∀ OntoVapourtec:holds.OntoLab:Vial
⊤ ⊑ ∀ OntoVapourtec:pumpsLiquidFrom.(OntoLab:ReagentBottle ⊔ OntoVapourtec:AutoSamplerSite)
⊤ ⊑ ∀ OntoVapourtec:recommendedReactionScale.om:Volume
⊤ ⊑ ∀ OntoVapourtec:sampleLoopVolume.om:Volume
⊤ ⊑ ∀ OntoVapourtec:toReceptacle.OntoLab:WasteBottle

Data properties:
∃ OntoVapourtec:lastLocalModifiedAt.⊤ ⊑ OntoVapourtec:VapourtecInputFile
∃ OntoVapourtec:lastUploadedAt.⊤ ⊑ OntoVapourtec:VapourtecInputFile
∃ OntoVapourtec:localFilePath.⊤ ⊑ OntoVapourtec:VapourtecInputFile
∃ OntoVapourtec:locationID.⊤ ⊑ (OntoVapourtec:AutoSamplerSite ⊔ OntoVapourtec:VapourtecR2Pump ⊔ OntoVapourtec:VapourtecR4Reactor)
∃ OntoVapourtec:remoteFilePath.⊤ ⊑ OntoVapourtec:VapourtecInputFile
⊤ ⊑ ∀ OntoVapourtec:lastLocalModifiedAt.xsd:decimal
⊤ ⊑ ∀ OntoVapourtec:lastUploadedAt.xsd:decimal
⊤ ⊑ ∀ OntoVapourtec:localFilePath.xsd:string
⊤ ⊑ ∀ OntoVapourtec:locationID.xsd:string
⊤ ⊑ ∀ OntoVapourtec:remoteFilePath.xsd:anyURI

OntoHPLC
Classes:
OntoHPLC:ChromatogramMeasurement ⊑ saref:Function

OntoHPLC:ChromatogramMeasurementCommand ⊑ saref:Command
OntoHPLC:ChromatogramPoint ⊑ = 1 OntoHPLC:atRetentionTime.OntoHPLC:RetentionTime
OntoHPLC:ChromatogramPoint ⊑ = 1 OntoHPLC:hasPeakArea.OntoHPLC:PeakArea
OntoHPLC:HighPerformanceLiquidChromatography ⊑ OntoLab:LabEquipment
OntoHPLC:InternalStandard ⊑ OntoCAPE_Phase_System:PhaseComponent
OntoHPLC:LiquidChromatography ⊑ saref:Task
OntoHPLC:PeakArea ⊑ om:Quantity
OntoHPLC:ResponseFactor ⊑ om:QuantityOfDimensionOne
OntoHPLC:RetentionTime ⊑ om:Duration

Object properties:
∃ OntoHPLC:atRetentionTime.⊤ ⊑ OntoHPLC:ChromatogramPoint
∃ OntoHPLC:characterises.⊤ ⊑ OntoHPLC:HPLCJob
∃ OntoHPLC:generatedFor.⊤ ⊑ OntoHPLC:HPLCReport
∃ OntoHPLC:hasJob.⊤ ⊑ OntoHPLC:HighPerformanceLiquidChromatography
∃ OntoHPLC:hasPastReport.⊤ ⊑ OntoHPLC:HighPerformanceLiquidChromatography
∃ OntoHPLC:hasPeakArea.⊤ ⊑ OntoHPLC:ChromatogramPoint
∃ OntoHPLC:hasReport.⊤ ⊑ OntoHPLC:HPLCJob
∃ OntoHPLC:hasResponseFactor.⊤ ⊑ OntoHPLC:HPLCMethod
∃ OntoHPLC:hasRetentionTime.⊤ ⊑ OntoHPLC:HPLCMethod
∃ OntoHPLC:indicatesComponent.⊤ ⊑ OntoHPLC:ChromatogramPoint
∃ OntoHPLC:records.⊤ ⊑ OntoHPLC:HPLCReport
∃ OntoHPLC:refersToSpecies.⊤ ⊑ (OntoHPLC:ResponseFactor ⊔ OntoHPLC:RetentionTime)
∃ OntoHPLC:reportExtension.⊤ ⊑ OntoHPLC:HighPerformanceLiquidChromatography
∃ OntoHPLC:usesInternalStandard.⊤ ⊑ OntoHPLC:HPLCMethod
∃ OntoHPLC:usesMethod.⊤ ⊑ OntoHPLC:HPLCJob
⊤ ⊑ ∀ OntoHPLC:atRetentionTime.OntoHPLC:RetentionTime
⊤ ⊑ ∀ OntoHPLC:characterises.OntoReaction:ReactionExperiment
⊤ ⊑ ∀ OntoHPLC:generatedFor.OntoLab:ChemicalAmount
⊤ ⊑ ∀ OntoHPLC:hasJob.OntoHPLC:HPLCJob
⊤ ⊑ ∀ OntoHPLC:hasPastReport.OntoHPLC:HPLCReport
⊤ ⊑ ∀ OntoHPLC:hasPeakArea.OntoHPLC:PeakArea
⊤ ⊑ ∀ OntoHPLC:hasReport.OntoHPLC:HPLCReport
⊤ ⊑ ∀ OntoHPLC:hasResponseFactor.OntoHPLC:ResponseFactor

$\top \sqsubseteq \forall$ OntoHPLC:hasRetentionTime.OntoHPLC:RetentionTime
$\top \sqsubseteq \forall$ OntoHPLC:indicatesComponent.OntoCAPE_Phase_System:PhaseComponent
$\top \sqsubseteq \forall$ OntoHPLC:records.OntoHPLC:ChromatogramPoint
$\top \sqsubseteq \forall$ OntoHPLC:refersToSpecies.OntoSpecies:Species
$\top \sqsubseteq \forall$ OntoHPLC:reportExtension.yago:WikicatFilenameExtensions
$\top \sqsubseteq \forall$ OntoHPLC:usesInternalStandard.OntoHPLC:InternalStandard
$\top \sqsubseteq \forall$ OntoHPLC:usesMethod.OntoHPLC:HPLCMethod

Data properties:
$\exists$ OntoHPLC:lastLocalModifiedAt.$\top \sqsubseteq$ OntoHPLC:HPLCReport
$\exists$ OntoHPLC:lastUploadedAt.$\top \sqsubseteq$ OntoHPLC:HPLCReport
$\exists$ OntoHPLC:localFilePath.$\top \sqsubseteq$ OntoHPLC:HPLCMethod
$\exists$ OntoHPLC:localFilePath.$\top \sqsubseteq$ OntoHPLC:HPLCReport
$\exists$ OntoHPLC:localReportDirectory.$\top \sqsubseteq$ OntoHPLC:HighPerformanceLiquidChromatography
$\exists$ OntoHPLC:remoteFilePath.$\top \sqsubseteq$ OntoHPLC:HPLCMethod
$\exists$ OntoHPLC:remoteFilePath.$\top \sqsubseteq$ OntoHPLC:HPLCReport
$\exists$ OntoHPLC:retentionTimeMatchThreshold.$\top \sqsubseteq$ OntoHPLC:HPLCMethod
$\exists$ OntoHPLC:unidentified.$\top \sqsubseteq$ OntoHPLC:ChromatogramPoint
$\top \sqsubseteq \forall$ OntoHPLC:lastLocalModifiedAt.xsd:decimal
$\top \sqsubseteq \forall$ OntoHPLC:lastUploadedAt.xsd:decimal
$\top \sqsubseteq \forall$ OntoHPLC:localFilePath.xsd:string
$\top \sqsubseteq \forall$ OntoHPLC:localReportDirectory.xsd:string
$\top \sqsubseteq \forall$ OntoHPLC:remoteFilePath.xsd:anyURI
$\top \sqsubseteq \forall$ OntoHPLC:retentionTimeMatchThreshold.xsd:float
$\top \sqsubseteq \forall$ OntoHPLC:unidentified.xsd:boolean

OntoGoal
Classes:
OntoGoal:DesignOfExperiment $\sqsubseteq$ OntoGoal:Step
OntoGoal:DesignOfExperiment $\sqsubseteq \forall$ OntoGoal:hasNextStep.OntoGoal:RxnExpExecution
OntoGoal:Goal $\sqsubseteq \geq 1$ OntoGoal:hasPlan.OntoGoal:Plan
OntoGoal:Goal $\sqsubseteq \leq 1$ OntoGoal:desiresGreaterThan.om:Quantity
OntoGoal:Goal $\sqsubseteq \leq 1$ OntoGoal:desiresLessThan.om:Quantity
OntoGoal:GoalSet $\sqsubseteq \geq 1$ OntoGoal:hasGoal.OntoGoal:Goal

OntoGoal:GoalSet $\sqsubseteq$ = 1 OntoGoal:hasRestriction.OntoGoal:Restriction
OntoGoal:Plan $\sqsubseteq \geq$ 1 OntoGoal:hasStep.OntoGoal:Step
OntoGoal:PostProcessing $\sqsubseteq$ OntoGoal:Step
OntoGoal:Result $\sqsubseteq$ = 1 OntoGoal:refersTo.om:Quantity
OntoGoal:RxnExpExecution $\sqsubseteq$ OntoGoal:Step
OntoGoal:RxnExpExecution $\sqsubseteq \forall$ OntoGoal:hasNextStep.OntoGoal:PostProcessing
OntoGoal:RxnOptPlan $\sqsubseteq$ OntoGoal:Plan
OntoGoal:Step $\sqsubseteq \geq$ 1 OntoGoal:canBePerformedBy.OntoAgent:Service

Object properties:
OntoGoal:desiresGreaterThan $\sqsubseteq$ OntoGoal:desires
OntoGoal:desiresLessThan $\sqsubseteq$ OntoGoal:desires
$\exists$ OntoGoal:canBePerformedBy.$\top \sqsubseteq$ OntoGoal:Step
$\exists$ OntoGoal:desires.$\top \sqsubseteq$ OntoGoal:Goal
$\exists$ OntoGoal:desiresGreaterThan.$\top \sqsubseteq$ OntoGoal:Goal
$\exists$ OntoGoal:desiresLessThan.$\top \sqsubseteq$ OntoGoal:Goal
$\exists$ OntoGoal:hasGoal.$\top \sqsubseteq$ OntoGoal:GoalSet
$\exists$ OntoGoal:hasNextStep.$\top \sqsubseteq$ OntoGoal:Step
$\exists$ OntoGoal:hasPlan.$\top \sqsubseteq$ OntoGoal:Goal
$\exists$ OntoGoal:hasRestriction.$\top \sqsubseteq$ OntoGoal:GoalSet
$\exists$ OntoGoal:hasResult.$\top \sqsubseteq$ OntoGoal:Goal
$\exists$ OntoGoal:hasStep.$\top \sqsubseteq$ OntoGoal:Plan
$\exists$ OntoGoal:hasStep.$\top \sqsubseteq$ OntoGoal:RxnOptPlan
$\exists$ OntoGoal:refersTo.$\top \sqsubseteq$ OntoGoal:Result
$\top \sqsubseteq \forall$ OntoGoal:canBePerformedBy.OntoAgent:Service
$\top \sqsubseteq \forall$ OntoGoal:desires.om:Quantity
$\top \sqsubseteq \forall$ OntoGoal:desiresGreaterThan.om:Quantity
$\top \sqsubseteq \forall$ OntoGoal:desiresLessThan.om:Quantity
$\top \sqsubseteq \forall$ OntoGoal:hasGoal.OntoGoal:Goal
$\top \sqsubseteq \forall$ OntoGoal:hasNextStep.OntoGoal:Step
$\top \sqsubseteq \forall$ OntoGoal:hasPlan.OntoGoal:Plan
$\top \sqsubseteq \forall$ OntoGoal:hasRestriction.OntoGoal:Restriction
$\top \sqsubseteq \forall$ OntoGoal:hasResult.om:Quantity
$\top \sqsubseteq \forall$ OntoGoal:hasStep.OntoGoal:Step
$\top \sqsubseteq \forall$ OntoGoal:hasStep.(OntoGoal:DesignOfExperiment $\sqcup$ OntoGoal:PostProcessing $\sqcup$ O
ntoGoal:RxnExpExecution)

$\top \sqsubseteq \forall$ OntoGoal:refersTo.om:Quantity

Data properties:

$\exists$ OntoGoal:cycleAllowance.$\top \sqsubseteq$ OntoGoal:Restriction

$\exists$ OntoGoal:deadline.$\top \sqsubseteq$ OntoGoal:Restriction

$\top \sqsubseteq \forall$ OntoGoal:cycleAllowance.xsd:int

$\top \sqsubseteq \forall$ OntoGoal:deadline.xsd:string

B.3 Agent UMLs

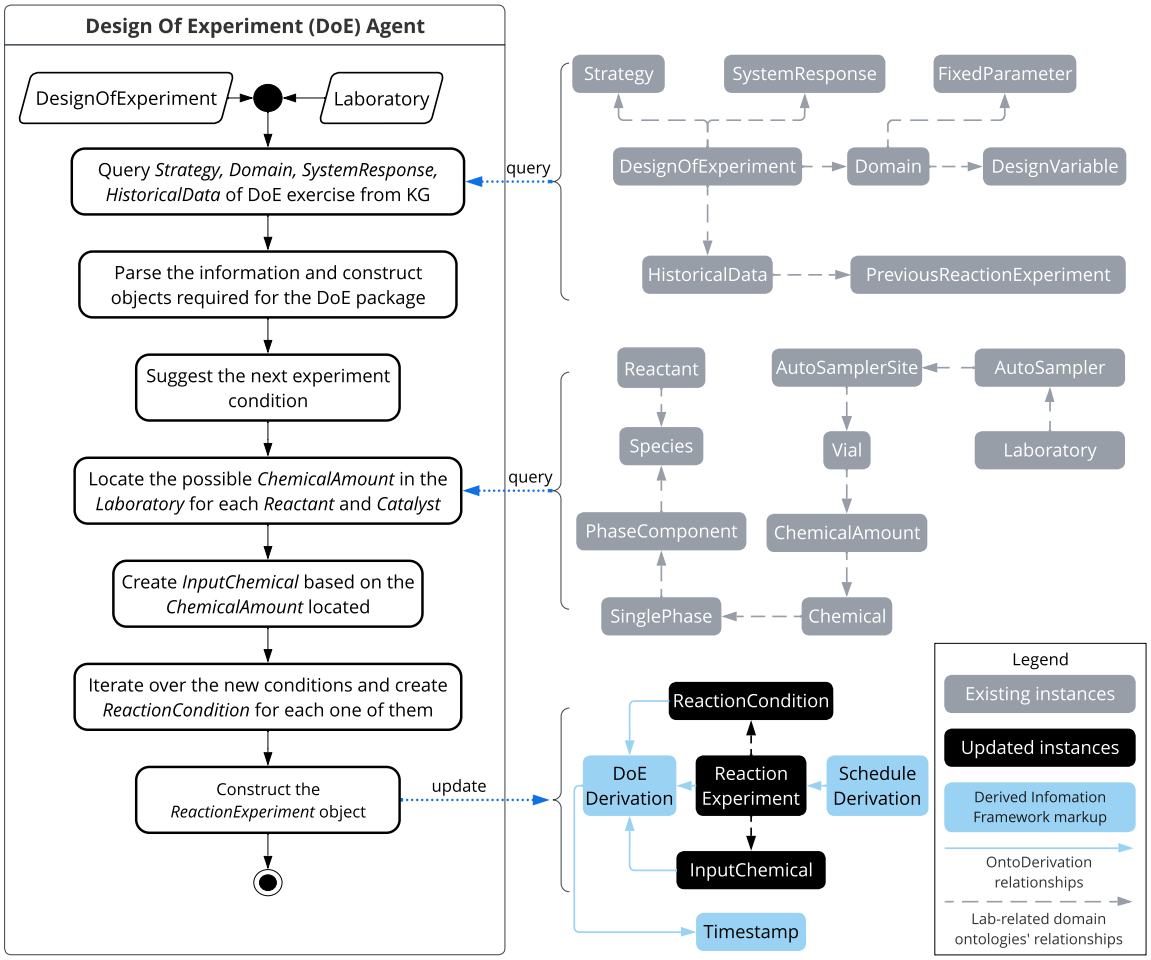


Fig. B.1 UML activity diagram of Design of Experiment (DoE) Agent. The blue arrows denote the data query and update between objects held in memory by the agent and instances in the knowledge graph.

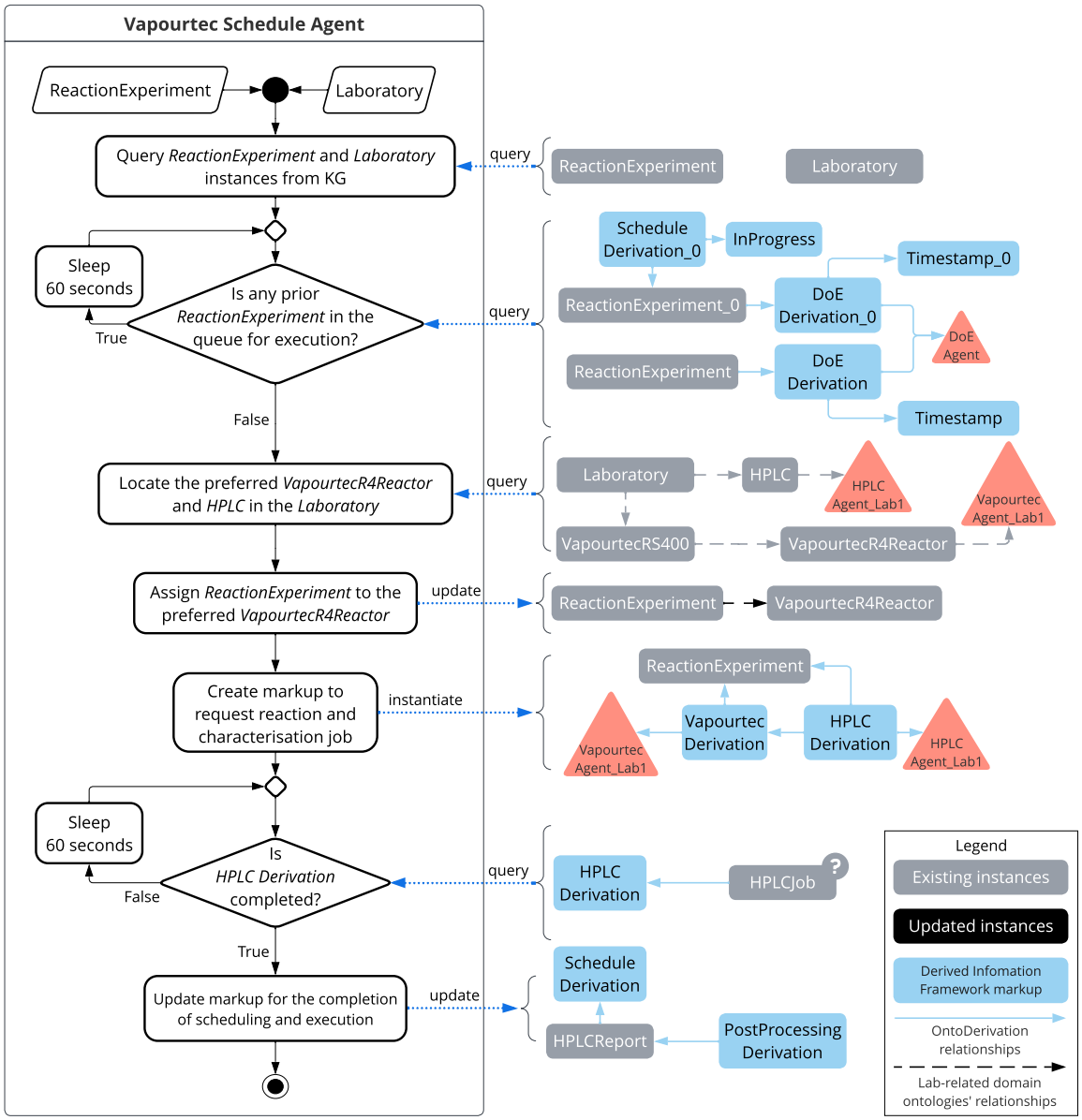


Fig. B.2 UML activity diagram of Vapourtec Schedule Agent. The blue arrows denote the data query and update between objects held in memory by the agent and instances in the knowledge graph. The question mark refers to the process of querying if the instance is available in the knowledge graph.

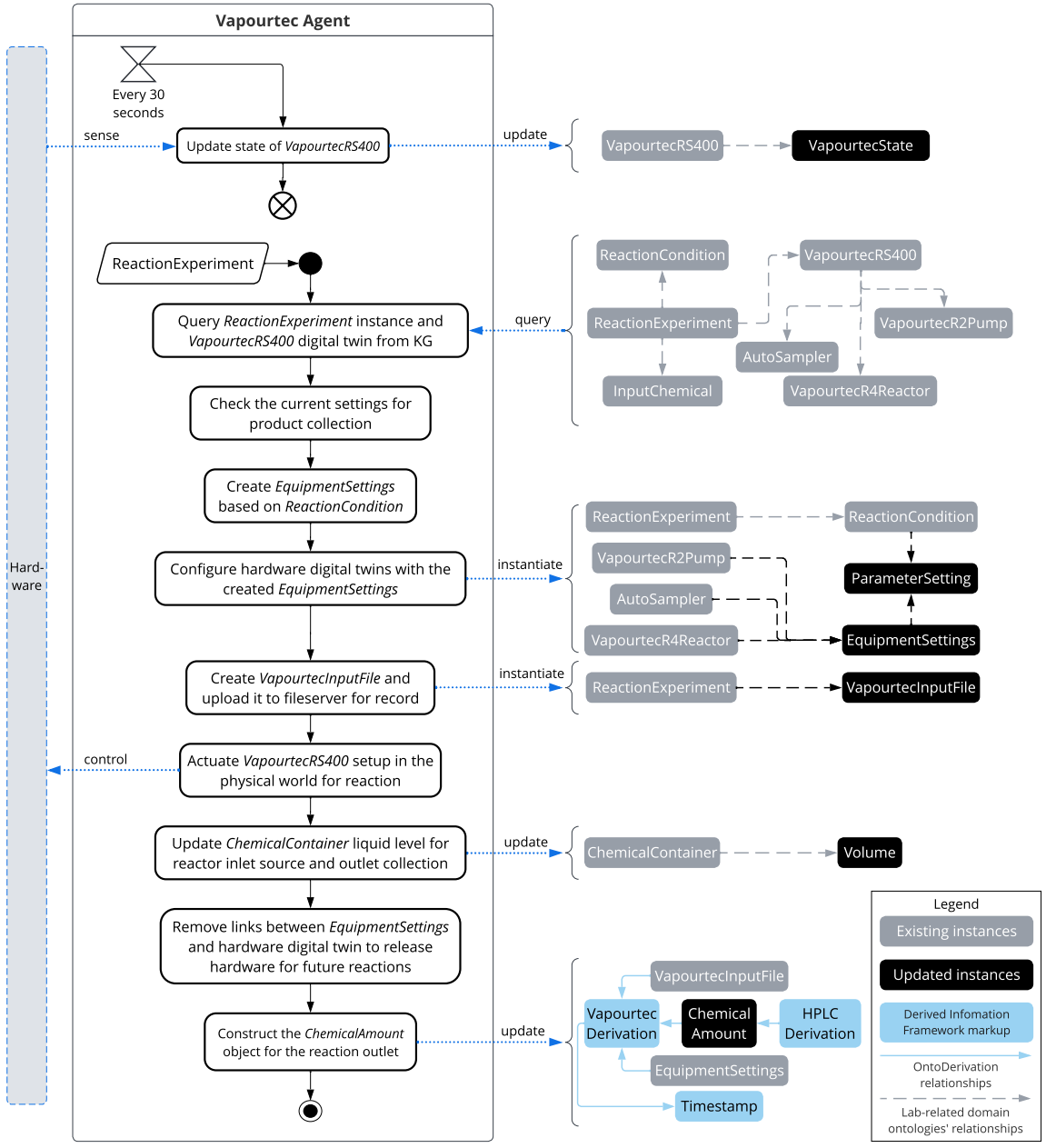


Fig. B.3 UML activity diagram of Vapourtec Agent. The blue arrows on the left-hand side refer to commands and data exchanged between the agent and the hardware to the agent, whereas those on the right-hand side denote the data query and update between objects held in memory of the agent and instances in the knowledge graph.

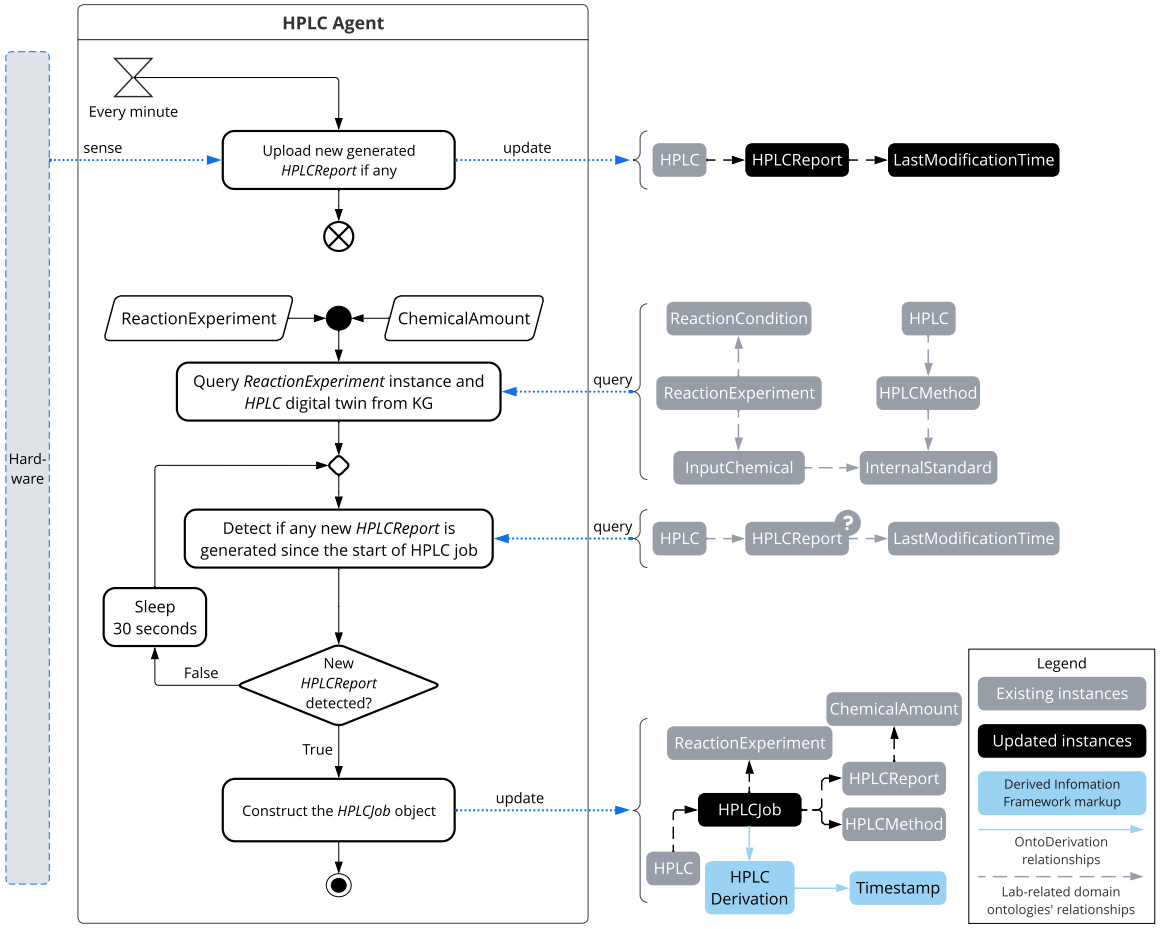


Fig. B.4 UML activity diagram of HPLC Agent. The blue arrow on the left-hand side refers to data transmitted from the hardware to the agent, whereas those on the right-hand side denote the data query and update between objects held in memory of the agent and instances in the knowledge graph. The question mark refers to the process of querying if the instance is available in the knowledge graph.

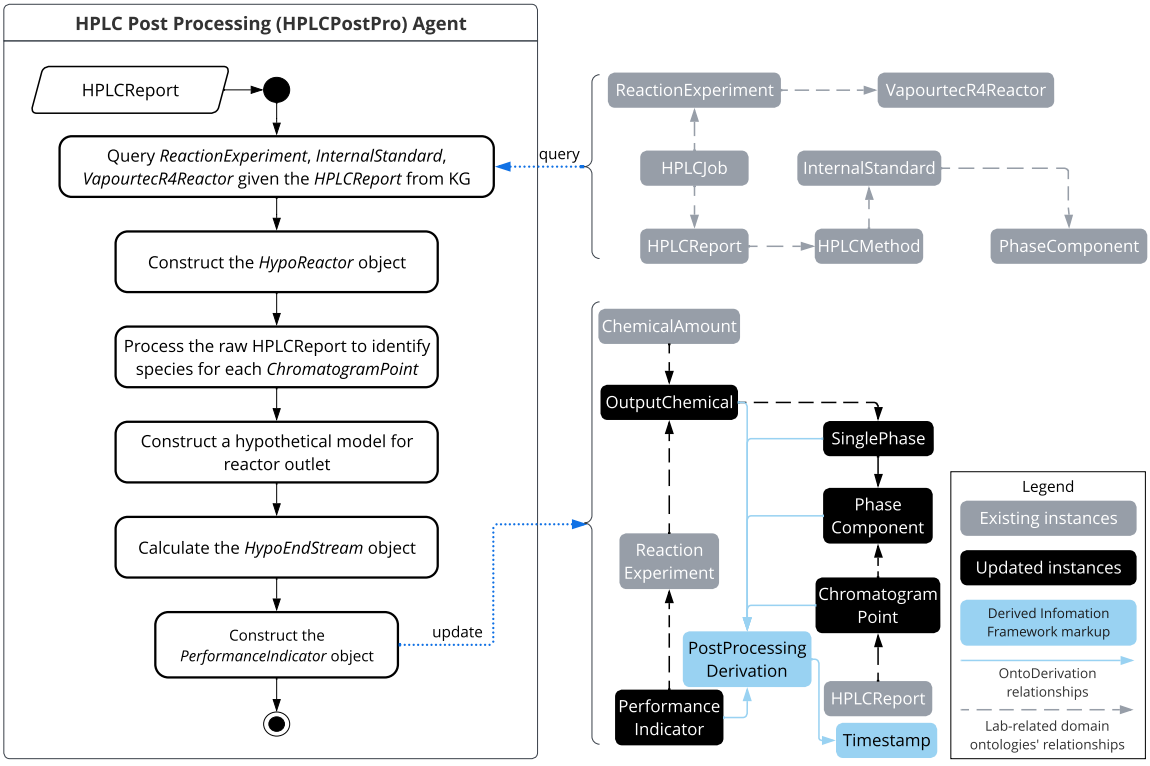


Fig. B.5 UML activity diagram of HPLC Post Processing (HPLCPostPro) Agent. The blue arrows denote the data query and update between objects held in memory by the agent and instances in the knowledge graph.

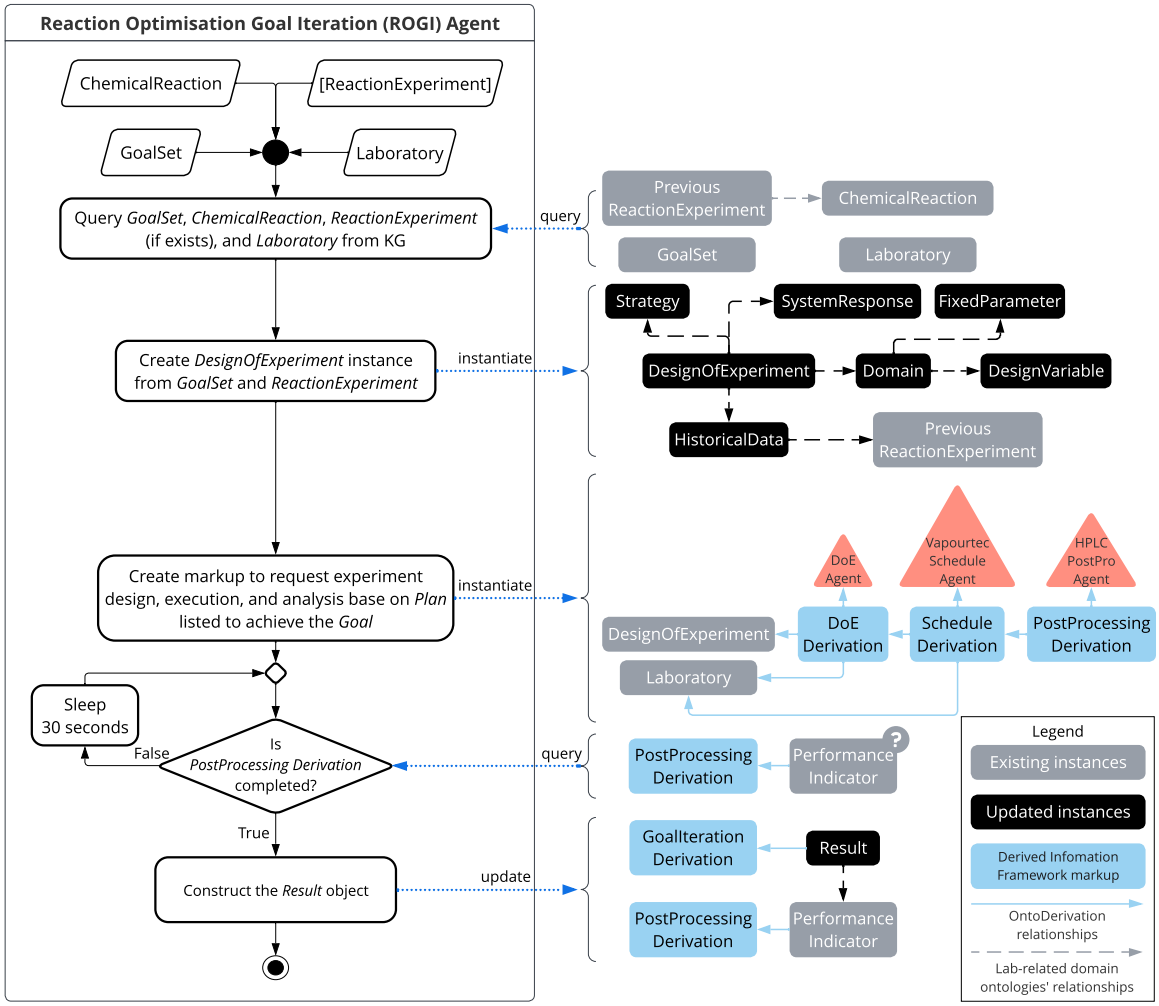
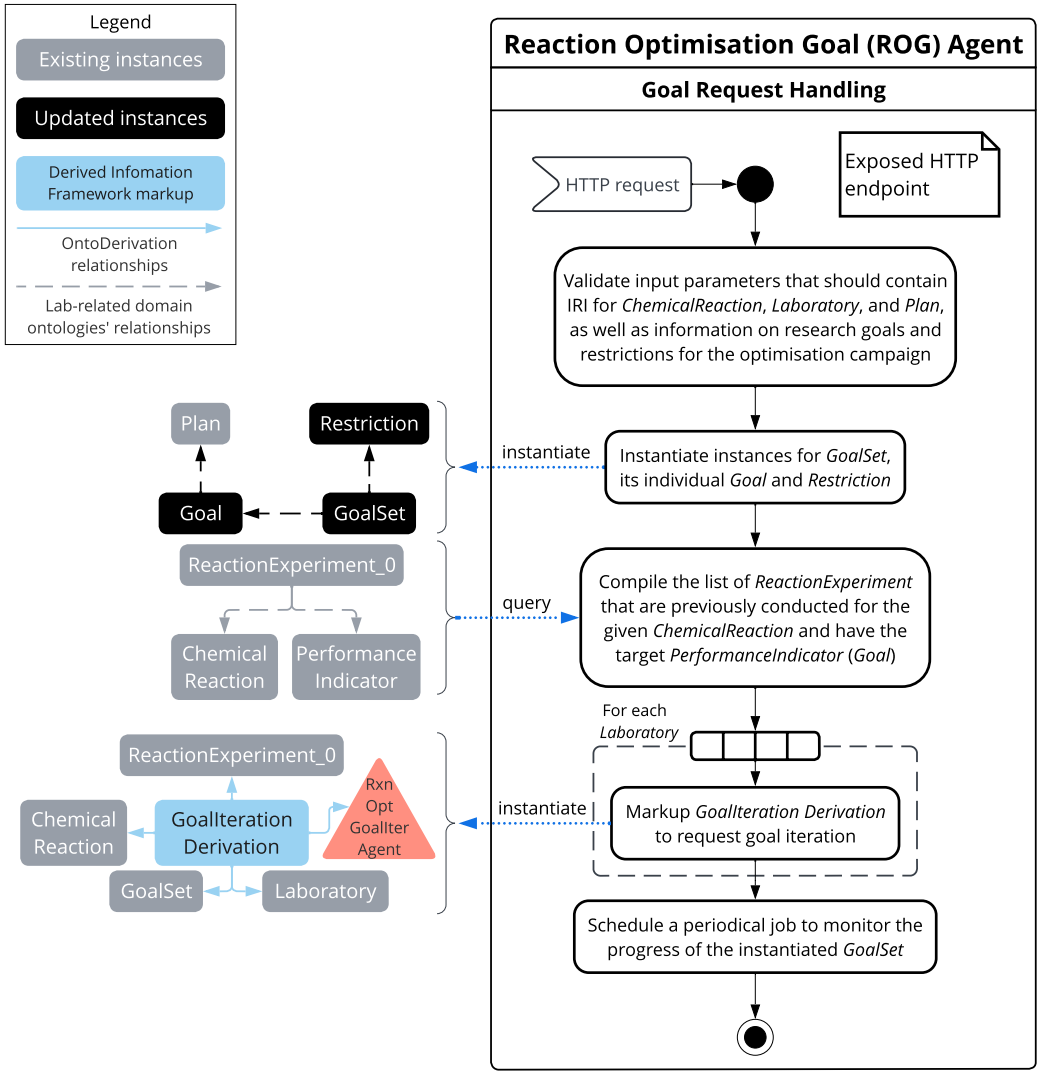
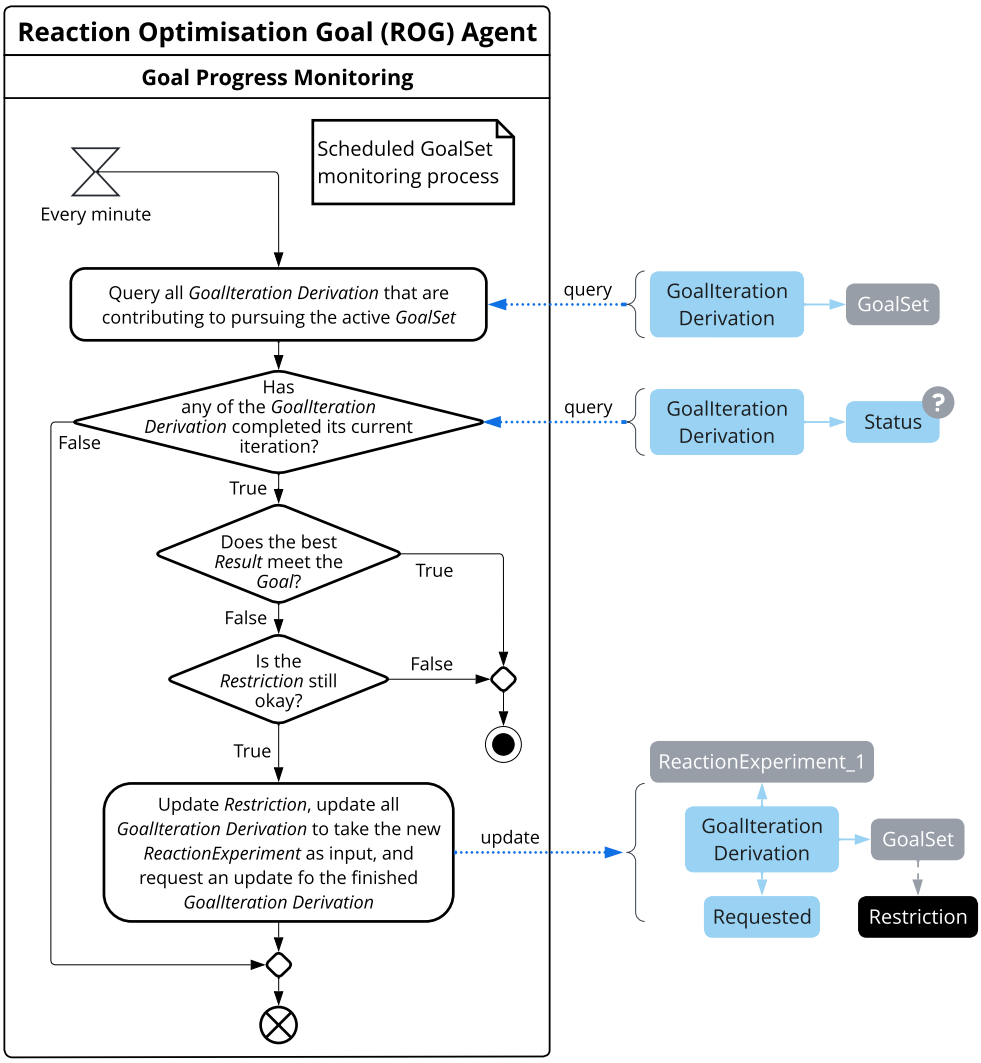


Fig. B.6 UML activity diagram of Reaction Optimisation Goal Iteration (ROGI) Agent. The blue arrows denote the data query and update between objects held in memory by the agent and instances in the knowledge graph. The question mark refers to the process of querying if the instance is available in the knowledge graph.



(a) The goal request handling occurs upon an HTTP request is issued from a web front end.



(b) The goal progress monitoring occurs as a periodical job following the processing of a goal request.

Fig. B.6 UML activity diagram of Reaction Optimisation Goal (ROG) Agent. The blue arrows denote the data query and update between objects held in memory by the agent and instances in the knowledge graph. The question mark refers to the process of querying if the instance is available in the knowledge graph.

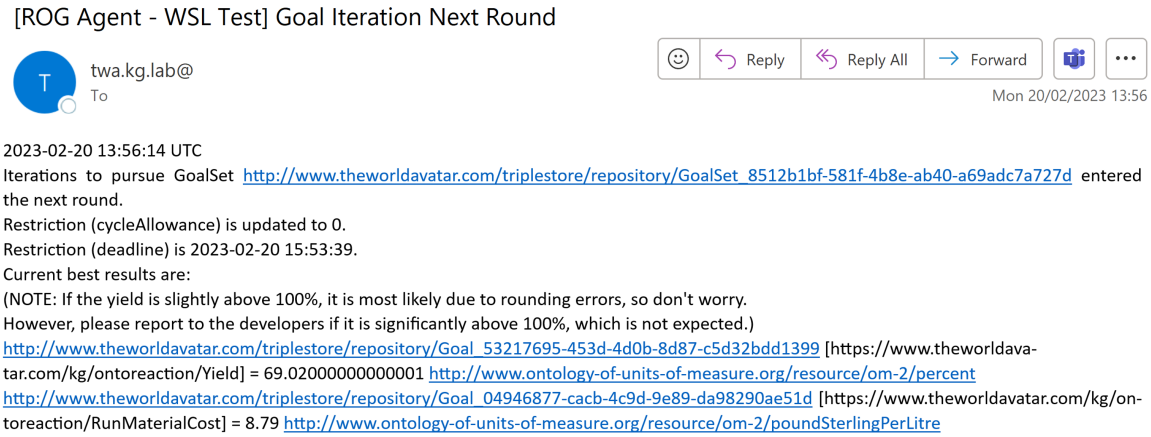


Fig. B.7 Example email notification for the progress of goal iteration.

Appendix C

Experimental details for distributed labs

This appendix provides a detailed account of the experimental results for the distributed labs use case. We begin by presenting the flow chemistry platforms involved in this work, followed by the HPLC calibration data and the cost of chemicals used to calculate the yield and cost objectives. Next, we report the results obtained under control conditions and finally present the results obtained during the self-optimisation campaign.

C.1 Flow chemistry platforms

The work in Chapter 6 involves two similar automated flow chemistry platforms located in Cambridge and Singapore. The method of sourcing input chemicals differs, with a liquid handler employed in Cambridge and reagent bottles utilised in Singapore. Brief descriptions of the experimental setups are provided below. All chemicals were used as received.

C.1.1 Cambridge lab

On the Cambridge side, the experimental setup consists of two Vapourtec R2 pump modules, one Vapourtec R4 reactor module, one Gilson GX-271 liquid handler, one four-way VICI switching valve (CI4W.06/.5 injector), and Shimadzu CBM-20A HPLC analytical equipment equipped with Eclipse XDB-C18 column (Agilent part number: 993967-902). To initiate the reaction, the liquid handler dispenses a 2 mL solution of 0.5 M benzaldehyde **1** dissolved in acetonitrile (with 0.06 M biphenyl as an internal standard) into the sample loop of pump A. Acetone **2** (50% v/v in acetonitrile) and 0.1 M NaOH **3** in ethanol are similarly loaded into sample loops for pump B and C. After being transferred by the switching valve, the product (benzylideneacetone **4**) is analysed using online HPLC. The HPLC analysis lasts 17 min, with a mobile phase consisting of an 80:20 (v/v) binary mixture of water and acetonitrile running at a rate of 2 mL min^{-1}. All compounds are detected at an absorption wavelength of 254 nm.

C.1.2 Singapore lab

On the Singapore side, the experimental setup consists of two Vapourtec R2 pump modules, one Vapourtec R4 reactor module, one 6-port 2-position VICI switch valve equipped with 60 nL sampling rotor, and an Agilent 1260 Infinity II system equipped with a G1311B quaternary pump, Eclipse XDB-C18 column (Agilent product number: 961967-302), and G1314F Variable Wavelength Detector (VWD). The input chemical for the reaction is sourced from three reagent bottles that are directly attached to the Vapourtec pumps: pump A contains 0.5 M benzaldehyde **1** in acetonitrile (with 0.05 M naphthalene as an internal standard), pump B contains 6.73 M acetone **2** in acetonitrile (50% v/v in acetonitrile), and pump C contains 0.1 M NaOH **3** in ethanol. The HPLC quaternary pump method for online HPLC described by Jeraal et al. [178] is used. However, the VWD wavelength was changed differently over the 8 min analysis time as follows: the absorption wavelength is 248 nm for the initial 6.05 min and then switched to 228 nm until the end of acquisition.

C.2 HPLC calibration for benzylideneactone

The concentration of actual product in Eq. 6.1, *i.e.*, Actual c_{product}, is calculated using the concentration of the internal standard (IS) as follows:

$$\text{Actual } c_{\text{product}} = \frac{1}{rf} \times \text{Actual } c_{\text{IS}} \times \frac{\text{Area}_{\text{product}}}{\text{Area}_{\text{IS}}}, \tag{C.1}$$

where rf (response factor) is the slope obtained from a linear calibration curve of the form $y = rf \times x$ with zero intercept:

$$\frac{\text{Area}_{\text{product}}}{\text{Area}_{\text{IS}}} = rf \times \frac{c_{\text{product}}}{c_{\text{IS}}} \tag{C.2}$$

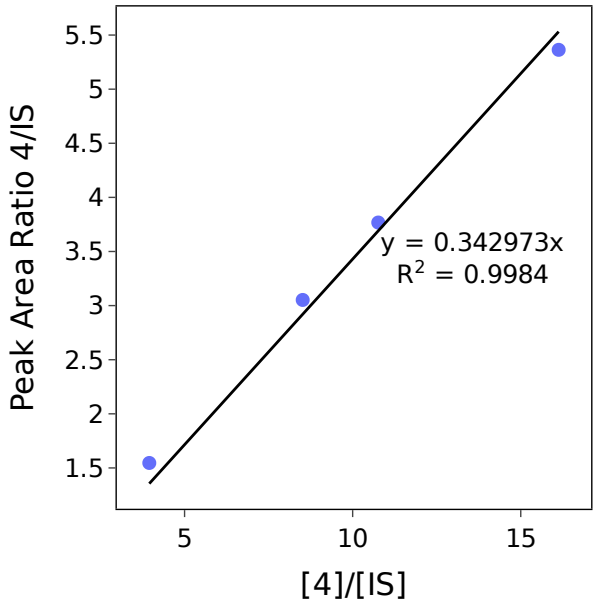
Details regarding the HPLC calibration in each lab are given below. Table C.1 and Fig. C.1(a) refer to the Cambridge lab where biphenyl is adopted as IS. Table C.2 and Fig. C.1(b) refer to the Singapore lab where naphthalene is adopted as IS.

Table C.1 HPLC calibration data for the analyte benzylideneacetone (4) with biphenyl as the internal standard in the Cambridge lab.

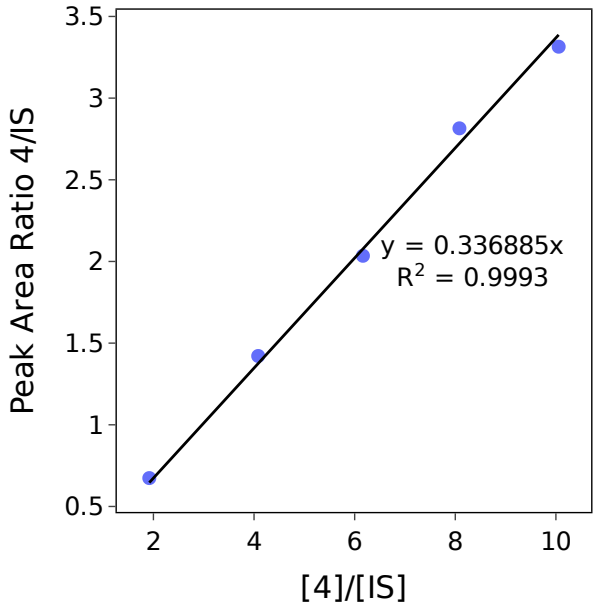
[4] (M)	[IS] (M)	[4]/[IS]	Peak Area 4	Peak Area IS	Peak Area Ratio 4/IS
0.289	0.018	16.13	20397712	3803050	5.36
0.208	0.019	10.76	15315627	4065088	3.77
0.155	0.018	8.51	12198929	3996727	3.05
0.075	0.019	3.95	6062851	3922075	1.55

Table C.2 HPLC calibration data for the analyte benzylideneacetone (4) with naphthalene as the internal standard in the Singapore lab.

[4] (M)	[IS] (M)	[4]/[IS]	Peak Area 4	Peak Area IS	Peak Area Ratio 4/IS
0.503	0.05	10.05	5317	1604	3.31
0.404	0.05	8.08	4901	1741	2.82
0.308	0.05	6.16	3628	1783	2.03
0.204	0.05	4.08	2628	1849	1.42
0.096	0.05	1.92	1295	1922	0.67



(a) Cambridge lab with biphenyl as the internal standard.



(b) Singapore lab with naphthalene as the internal standard.

Fig. C.1 HPLC calibration curve for the analyte benzylideneacetone (**4**) in two labs.

C.3 Cost of chemicals

Table C.3 lists the costs of reactants, catalyst, and solvents used to calculated the cost objective with Eq. 6.3. These costs are obtained on the Cambridge side, while all chemicals used on the Singapore side are purchased from Sigma-Aldrich.

Table C.3 Cost for chemicals used in the objective calculation.

Chemical	**Quantity**	**Cost (£ L^{-1})**
Benzaldehyde	£83.60 / 2.5 L[a]	33.44
Acetone	£28.45 / 1 L[b]	28.45
NaOH	£39.90 / 500 g[c]	169.97[d]
Acetonitrile	£127.00 / 1 L[e]	127
Ethanol	£327.5 / 2.5 L[f]	131

All below links were accessed on 21 Mar 2023.

[a] *https://www.sigmaaldrich.com/GB/en/product/sial/b1334.*

[b] *https://www.fishersci.co.uk/shop/products/acetone-laboratory-reagent-99-5-honeywell-8/15691640#?keyword=67641.*

[c] *https://www.fishersci.co.uk/shop/products/sodium-hydroxide-white-pellets-fisher-bioreagents/10192863#?keyword=1310732*

[d] *Calculated using density 2.13 kg L^{-1} from https://pubchem.ncbi.nlm.nih.gov/compound/14798#section=Density&fullscreen=true.*

[e] *https://www.fishersci.co.uk/shop/products/acetonitrile-hplc-fisher-chemical-8/10754361?searchHijack=true&searchTerm=A%2F0626%2F17&searchType=RAPID&matchedCatNo=A%2F0626%2F17.*

[f] *https://uk.vwr.com/store/product/7331577/ethanol-absolute-99-8-analar-normapur-acs-reag-ph-eur-analytical-reagent.*

C.4 Reproducibility across laboratories

Table C.4 presents the results for two control conditions from both laboratories to test reproducibility.

Table C.4 Control reactions and reproducibility for the lab in Cambridge and Singapore.

Equiv. 2	Equiv. 3	Res. Time (min)	Temperature (°C)	Yield (%)	Lab
22.5	0.12	10	50	70.94	Cambridge
				71.44	
				71.35	
				70.62	Singapore
				72.61	
				71.43	
14.92	0.16	8.3	69	59.92	Cambridge
				60.43	Singapore

C.5 Self-optimisation campaign

Table C.5 lists the boundaries of the continuous variables involved in the self-optimisation campaign. These ranges cover most of the design space explored by Jeraal et al. [178] with both molar equivalents adjusted to ensure reliable pump flow rates in the Cambridge lab.

Table C.5 Lower and upper limits used for the continuous variables in the self-optimisation campaign.

Limits	Equiv. 2	Equiv. 3	Res. Time (min)	Temperature (°C)
Lower	5	0.05	5	30
Upper	40	0.2	15	70

Table C.6 lists all experiments executed during the self-optimisation campaign.

Table C.6 Summary of the experiments recorded in the knowledge graph during the optimisation campaign in chronological order. For runs conducted in the Cambridge lab, the autosampler sites from which the chemicals were sourced are also provided.

Reaction Partial IRI	Equiv. 2	Equiv. 3	Res. Time (min)	Temp. (°C)	Yield (%)	Cost (£ L^{-1})	Equiv. 1 Site	Equiv. 2 Site	Equiv. 3 Site	Lab
2a39187f-d2e3-4d89-819b-a980f83f9c5a*	11.08	0.06	13.78	66	50.24	225.19	2	14	15	Cambridge
412135fc-a3ca-4336-a50b-42a7498d58f9**	22.5	0.12	10	50	70.7	330.58				Singapore
2cf1175d-d7b9-4d5a-9315-8934fcf49b68	22.5	0.12	10	50	73.18	330.58				Singapore
473233f1-5ad7-4f1e-ad21-91e567bbf826	7.3	0.12	5.38	69	47.42	242.66				Singapore
b7f43e93-678f-4e06-ba4c-02a22b91b759	22.58	0.18	11.6	55	73.93	370.38				Singapore
0c015baa-942e-4075-aaef-6bba8627a141	30.35	0.12	12.91	60	93	375.99	1	14	21	Cambridge
9b02806a-006f-4b4c-9923-446117c34ef8	36.4	0.1	13.87	42	67.66	397.87				Singapore
743b9664-d886-4dda-82d1-09ae2df4a6f4	17.65	0.17	5.29	47	82.08	335.31				Singapore
3e87773f-5960-4c83-8139-070c296cc047	5.06	0.06	13.73	55	65.28	190.36	1	14	21	Cambridge
b07ebae5-2b76-4c1e-8075-a773db5f8ae2	14.21	0.08	10.73	53	60.61	256.4				Singapore
60f02bf0-97b9-430b-acde-451521776670	8.5	0.15	15	45	56.69	269.27				Singapore
a85a4029-3692-4461-8f92-9891d7c171e0	38.43	0.08	5.56	34	41.22	396.5	1	8	19	Cambridge
fa86bdfa-b900-40e5-abbb-16e1a96d8d03	7.97	0.05	10.17	49	0	200.64				Singapore
23d7f9c3-0f0a-45d6-813f-35cd2a60e722	15.86	0.14	9.95	69	68.22	305.28	1	8	19	Cambridge
b0dda4a0-0469-4120-abe6-a41c6ea601db	35.7	0.18	13.12	63	52.76	446.27				Singapore
e957ae0a-f48c-4cc1-9ceb-c2eacc219932	11.57	0.05	5	50	71.01	221.47				Singapore
61ebe163-2268-4f6c-8539-9d05372f6dec	22.78	0.06	13.07	60	89.53	292.86	1	8	19	Cambridge
cb5fe529-8f7b-4d39-8232-488ad06b71a6	7.83	0.16	5.42	31	38.71	271.95				Singapore
3c72a62a-16dd-46d5-a60e-e44fcab37485	8.62	0.05	8.11	62	52.56	204.4				Singapore
44d86f92-6ece-42c6-b468-9b394e868e8c	12.81	0.06	7.61	66	89.6	235.19	1	8	19	Cambridge
3ad34730-dd25-4894-9ad8-ecad05591cde	5	0.07	9.2	62	29.43	196.57				Singapore
4059a808-1401-4514-ab3e-048b2295e227	5	0.1	13.89	33	61.14	216.24	1	8	19	Cambridge
1245c932-7945-4cad-959b-633c407d1927	9.1	0.09	13.86	56	*3558.59*†	233.4				Singapore
bcd0150a-0e1c-4275-bbfb-3329272a8dee	8.41	0.2	10.3	69	66.38	301.53	1	8	19	Cambridge
c04c0ffb-9f2b-4e8a-ac17-07379ad776ef	6.07	0.05	7.1	45	22.15	189.65	1	8	19	Cambridge
13e12d18-8b6a-4e3b-ad07-de3bbff82370	5	0.05	12.74	33	17.4	183.46	1	8	19	Cambridge
205fedf7-8b45-4aa3-a71f-28b1d940062c	5.82	0.1	14.2	69	59.59	220.98	1	8	19	Cambridge
5cabcc77-f225-4bb8-a302-aad5b55e112c	35.89	0.05	15	55	41.37	362.14	1	8	19	Cambridge
1a5ec07d-5081-403a-ad70-1e8b118fbb23	15.18	0.05	8.65	68	33.95	242.35	2	12	18	Cambridge
48a7ca2f-dd1c-48fb-89a8-3a99e74162a1	5	0.07	5	34	1.19	196.57	2	12	18	Cambridge

Continued

Table C.6 (Continued)

Reaction Partial IRI	Equiv. 2	Equiv. 3	Res. Time (min)	Temp. (°C)	Yield (%)	Cost (£ L^{-1})	Equiv. 1 Site	Equiv. 2 Site	Equiv. 3 Site	Lab
90db1c6a-b000-4d4b-98d2-a2008587f196	19.43	0.07	13.18	64	87.39	280.04	6	12	18	Cambridge
e766d2c7-e15a-4fc3-8bf9-0f04ac3bad04	20.26	0.05	10.5	69	71.26	271.73	6	12	18	Cambridge
c765eb8c-c677-48ed-b631-9c727d22af33	16.24	0.07	9.11	30	34.14	261.59	6	12	18	Cambridge
0d6d3049-4582-4f41-8068-6b974845d027	7.26	0.06	5	68	44.04	203.09				Singapore
0c805777-0eac-4f77-b29b-a01022686b30	27.29	0.05	9.92	62	34.85	312.4	6	14	16	Cambridge
3d0fb333-f192-4763-ae4b-500711630692	5	0.06	8.04	69	61.64	190.02	6	14	16	Cambridge
8c0d6a77-c601-45eb-840a-4a1065729ac5	17.99	0.1	14.64	61	71.67	291.38	6	14	16	Cambridge
92151378-b70c-4907-99ca-2c0ccf839831	18.19	0.05	12.25	61	76.88	259.76	6	14	16	Cambridge
60953e65-1fe7-4666-bfbf-d3d5f5c9ba93	13.41	0.12	11.8	61	68	278	6	14	16	Cambridge
a07840d1-000d-457d-b198-67355065a4fe	19.23	0.05	13.5	63	$0^{\dagger\dagger}$	265.77	6	14	16	Cambridge
6f6514cb-299c-4686-a5fe-4755facdb2f1	5.08	0.2	12.61	49	59.59	282.26	6	14	16	Cambridge
b4a438bf-3fe0-44de-8dc3-f54a28ba0564	5	0.05	15	30	6.86	183.46	6	14	16	Cambridge
632f3d6c-e258-4051-99f7-5dc9a2ee3499	5	0.08	13.71	43	15.98	203.13	6	13	16	Cambridge
65a3db4b-9bdf-40dd-8f24-277eb599e5d7	5	0.05	6.15	30	2.75	183.46	7	13	16	Cambridge
3f28b2cf-dbb9-4426-99a4-a980f5bf2d9d	19.06	0.05	15	61	76.34	264.79	7	13	16	Cambridge
76aa4623-1747-4802-b9bf-f57dd79f047e	8.36	0.16	12.17	56	67.19	275.01	7	13	16	Cambridge
d42cd029-7be7-42bb-ae37-dc10e4f700a9	17.65	0.15	14.46	67	57.4	322.19	7	13	16	Cambridge
5ead68d1-9b19-4685-b602-f3329cc3b3bf	5	0.05	15	52	40.53	183.46	7	13	21	Cambridge
c2cb9ad0-42e2-4f4e-81ca-9058a6498812	23.93	0.09	15	62	82.83	319.18	5	12	17	Cambridge
c31c4dea-6edb-44bc-9e61-406b7db8e4ac	18.5	0.05	5	68	47.28	261.55	5	12	17	Cambridge
917da8be-6620-4cc7-b1df-95cb3d78ad49	22.29	0.1	12.37	58	72.64	316.25	5	12	17	Cambridge
52321782-ca81-4f7e-8296-c38ea2731e93	21.08	0.15	15	46	73.15	342.03	5	12	17	Cambridge
38323622-6232-40e0-b8ee-80b21466eb99	5.02	0.17	8.35	54	56.75	262.25	5	12	17	Cambridge
cd283f42-f882-43bc-ad3f-cf20896a0a0b	12.16	0.12	7.16	69	62.44	270.77	5	12	17	Cambridge
73f449aa-34ff-47d0-aa39-052d3cad34fb	5	0.11	14.69	69	47.93	222.8	5	12	17	Cambridge
c477e3d6-eea5-454c-b1f1-2c6ee528c05f	21.49	0.12	12.97	56	66.08	324.74	5	8	17	Cambridge
9c7bdf7c-b090-4896-afa6-fec99ab6e808	5.72	0.05	13.38	69	58.75	187.63	5	8	17	Cambridge
a85b964a-1e3b-4e45-8080-f64215698d01	17.83	0.07	14.54	48	32.41	270.79	5	8	17	Cambridge
e02256cd-dffe-4645-8080-dde8eb2dad19	5	0.09	15	66	48.93	209.69	5	13	17	Cambridge
50872203-0a25-42db-b8c5-aee440e7263c	5.63	0.07	12.9	57	27.28	200.22	5	13	17	Cambridge
8c8e54eb-4e73-4ddf-9a6d-abfd9ce8e299	22.52	0.12	9.85	65	70.59	330.7	4	13	18	Cambridge

Continued

Table C.6 (Continued)

Reaction Partial IRI	Equiv. 2	Equiv. 3	Res. Time (min)	Temp. (°C)	Yield (%)	Cost (£ L^{-1})	Equiv. 1 Site	Equiv. 2 Site	Equiv. 3 Site	Lab
b854e692-f6f4-494a-b6ac-f2103f4bda7f	8.23	0.05	15	69	69.21	202.14	4	13	18	Cambridge
98d72768-16d0-4a22-a89f-86c4ff9741d5	14.45	0.13	6.28	60	72.69	290.57	4	13	18	Cambridge
298eff02-90e0-42a1-95b7-e8d307547606	19.19	0.13	12	45	72.57	317.99	4	13	18	Cambridge
80a25214-4ea7-4a35-b255-f2042b75f16e	5	0.05	10.41	69	58	183.46	4	13	18	Cambridge
c253af7f-3bd6-43c5-9dcc-9d1c89dc5846	8.6	0.07	5	66	40.61	217.4	4	11	18	Cambridge

*The complete IRI starts with `https://www.theworldavatar.com/kg/lab_auto/derivation/ReactionExperiment_`

**The complete IRI for all of the remaining rows starts with `https://www.theworldavatar.com/kg/lab_auto/derivation/ReactionVariation_`

†This yield is considered abnormal and therefore not utilised by the DoE algorithm, *i.e.*, not included in the interactive animation. The correct yield should be 41.43%, which is still dominated.

††Unfortunately, this wrong yield is utilised by the DoE algorithm. The correct yield should be 71.22%, which is still dominated.