Data Science 1 (2016) 1–5
IOS Press

Towards a scientific data framework to support scientific model development

New requirements emerging from the combustion kinetics domain

Gabriele Scalia [a,*], Matteo Pelucchi [b], Alessandro Stagni [b], Alberto Cuoci [b], Tiziano Faravelli [b] and Barbara Pernici [a]

[a] *Department of Electronics, Information and Bioengineering, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20131 Milano, Italy*
E-mails: gabriele.scalia@polimi.it, barbara.pernici@polimi.it
[b] *Department of Chemistry, Materials, and Chemical Engineering Giulio Natta, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20131 Milano, Italy*
E-mails: matteo.pelucchi@polimi.it, alessandro.stagni@polimi.it, alberto.cuoci@polimi.it, tiziano.faravelli@polimi.it

Abstract. The sharing of scientific and scholarly data has been increasingly promoted over the last decade, leading to open repositories in many different scientific domains. However, data sharing and open data are not final goals in themselves, the real benefit is in data reuse, which allows leveraging investments in research and enables large-scale data-driven research progress. Focusing on reuse, this paper discusses the design of an integrated framework to automatically take advantage of large amounts of scientific data extracted from the literature to support research, and in particular scientific model development. Scientific models reproduce and predict complex phenomena and their development is a rather challenging task, within which scientific experiments have a key role in their continuous validation. Starting from the combustion kinetics domain, this paper discusses a set of use cases and a first prototype for such a framework which leads to a set of new requirements and an architecture that can be generalized to other domains. The paper analyzes the needs, the challenges and the research directions for such a framework, in particular those related to data management, automatic scientific model validation, data aggregation and data analysis, to leverage large amounts of published scientific data for new knowledge extraction.

Keywords: scientific model development, experimental data, scientific data, scholarly data, scientific model validation, combustion kinetics

1. Introduction

Scientific repositories allow the collection and distribution of scientific and scholarly data. Among the fundamental factors contributing to their growth is an increasing availability of computing power and storage capacity as well as new *big data* analysis techniques to manage such vast amounts of data. However, since their initial establishment, some challenges have been highlighted in the literature, such as availability, integration, extensibility and maintainability [1].

The urgent need of improving the infrastructure supporting the *reuse* of experimental data has been highlighted in the literature and led to general guidelines to create and manage repositories, notably the

*Corresponding author. E-mail: gabriele.scalia@polimi.it.

FAIRness [2] (being findable, accessible, interoperable and reusable) or the "pyramid" of needs for data management that span from being simply *saved* to being *shared* to ultimately being *trusted* [3]. The collection of scientific data is becoming more and more important for the validation of research results. One of the current goals is to improve the capability of sharing experimental data among researchers, in order to enhance their quality and reproducibility and to derive new research results. Based on the Open Science Cloud strategy of the EU[1], *every* project financed within the H2020 framework has to comply with the FAIR data policy.

Sharing of scientific data has been increasingly promoted over the last decade. However, it was recently discussed how "data sharing practices, especially motivations and incentives, have received far more study than has data reuse" and that *data sharing* and *open data* are not final goals in themselves, but the real benefit is in *data reuse*, which is "an understudied problem that requires much more attention if scientific investments are to be leveraged effectively" [4]. Reuse leverages investments in research and enables large-scale data-driven research progress. However, in order to reuse datasets from multiple sources, challenges, such as integration, must be addressed.

This paper focuses on reuse, and in particular on the design of an integrated system to automatically take advantage of large amounts of scientific experimental data with the goal to support new research. This requires the integration of data management and analysis techniques in the scientific development workflow. If a scientific repository with its tools can improve the efficiency of many tasks, such as searching and exporting selected entries, usage of new data analysis techniques can leverage the large amounts of integrated scientific data to automatically and dynamically discover knowledge normally not obtainable manually or through partial datasets given the large volumes of data required. We consider experimental data published in scientific publications, which are typically delivered as attached supplementary materials or made available in scientific and scholarly data repositories.

One key research activity in many scientific domains is *scientific model development*, where the goal is to build models to reproduce and predict complex phenomena. Experimental data, typically extracted from scientific publications, have a crucial role within this activity. Indeed, extensive comparison of model simulations with real experimental data is needed in the iterative scientific model development life cycle, to continuously validate and improve the scientific models being developed.

In this paper, we focus on experimental data and model development in the *combustion kinetics* domain, where a number of initiatives to collect experimental data in a systematic way have already been developed [5–8], along with initiatives to define better community data reporting standards[2]. Even if this paper focuses on a specific domain, the use cases, the requirements and the solutions analyzed are common to many other scientific domains, as discussed in Section 3.

Automatically taking advantage of published scientific data to aid model development sets several challenges:

- First of all, data not only need to be collected, but also *semantically interpreted* and *integrated* through domain-specific ontologies. The consideration of the semantics of stored data requires "machines to be capable of autonomously and appropriately acting when faced with the wide range of types, formats, and access-mechanisms/protocols that will be encountered during their self-guided exploration" [2].
- *Data quality* must be ensured since it has an impact on the whole model development cycle and low data quality could bring wrong results. *Data cleaning* is an important activity in this respect.

[1] https://ec.europa.eu/research/openscience/index.cfm?pg=open-science-cloud
[2] http://www.smartcats.eu/wg4/task-force/

- Objective and *quantitative measures* to assess the performance of a model with respect to some experimental data are necessary (e.g. [9]).
- *Dynamically evolving sources of data* must be taken into account for complex analysis tasks.

Starting from a description of the most important *use cases*, which also emerged from previous works [10, 11], this paper describes the development of a *new prototype* which goes in the direction of supporting them, the *new requirements* that emerged from the prototype development and an *architecture* which, while also taking new requirements into account, outlines the proposed framework's future development.

Therefore, this paper's novel contributions are as follows

- Several use cases for an integrated system which leverages large amounts of published scientific data extracted from heterogeneous sources to support scientific model development are presented.
- A new prototype is developed and distributed.
- New system requirements, which emerge also from the prototype development, are identified and discussed.
- A data-based and service-based architecture for such a system is outlined and discussed, taking into account the new requirements and focusing in particular on the data model and the design of a set of data curation and analysis services.

This paper is an extension of [11], presented at the SAVE-SD workshop on *Semantics, Analytics, Visualisation: Enhancing Scholarly Dissemination*. This paper shares with it the analysis of the combustion kinetics scenario and the first generalized use cases and requirements arising from it. However, with respect to the previous work, the use cases have been refined and detailed, a completely new prototype has been developed, and this led to new requirements and an architectural framework design which has been heavily improved and detailed with respect to those presented in [11].

The paper is structured as follows: Section 2 introduces the domain and the general approach to scientific model development, Section 3 discusses related work, Section 4 describes a set of use cases for the system and the developed extended prototype, Section 5 discusses the emerging requirements and Section 6 outlines an integrated architectural framework to support them. Finally, Section 7 discusses perspectives for future research directions.

2. Scenario

In this section, we describe the scenario — kinetic modelling of combustion processes. Kinetic models determine the reactivity of a given fuel or a fuel mixture, and combustion kinetic modelling has been driving the development of more efficient fuels and combustion technologies for the last 40 years.

The development and update of reliable kinetic models is a rather challenging task, directly reflecting the intrinsic complexity of combustion phenomena, and it is one of the fields of research of the CRECK modeling group[3]. Such models typically involve $\sim 100-1,000$ chemical species connected by a network of $\sim 1,000 - 10,000$ elementary reaction steps. Moreover, a combustion kinetic model hierarchically develops from small chemical species (e.g., hydrogen, methane, and so on) up to heavier compounds typically found in commercial fuels such as gasoline, diesel, jet and alternative fuels. For this reason, any modification in the core model significantly propagates its effects to heavier species making continuous revisions and updates mandatory to preserve the model's reliability.

[3] http://creckmodeling.chem.polimi.it/

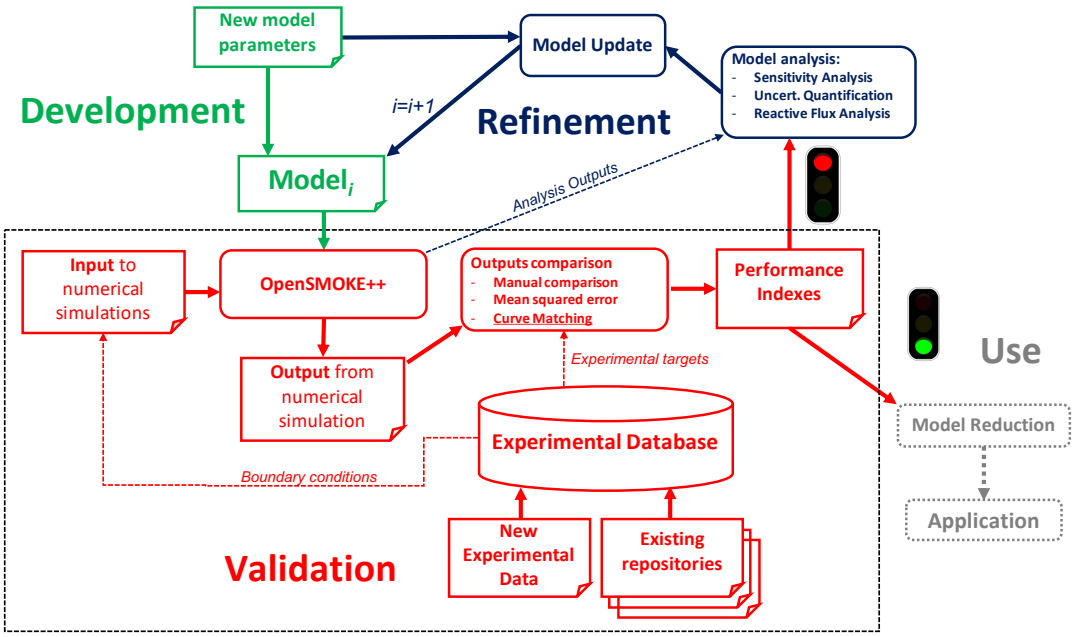

Fig. 1. Standard development, validation and refinement cycle of a chemical kinetic model for combustion applications. The process is represented as a flow of *data* and *activities* and has multiple starting points. Dashed lines represent auxiliary data flows. In this paper, we mainly focus on the *validation* phase (dotted rectangle). A validation step could be triggered by the development of a new model — which in turn could be the result of the *refinement* of an existing one — or by the acquisition of new experimental data in the database.

OpenSMOKE++ [12] is an example of software developed to execute such models performing kinetic simulations of typical facilities such as jet stirred and flow reactors, 1D-2D laminar flames, shock tubes and rapid compression machines, specifically conceived to decouple purely chemical kinetic effects from fluid dynamics, heat and mass transfer phenomena. Variables of interest are typically ignition delay times, laminar flame speeds, fuel/oxidizer mixtures, fuel consumption, and intermediate product species formation/disappearance at specific conditions of temperature, pressure and dilution.

The standard development, validation and refinement cycle of chemical kinetic models for combustion applications is shown in Figure 1. From an operational perspective, the iterative *validation* of such models (dotted rectangle) strongly relies on extensive comparisons of results from model simulations with experimental data covering conditions of interest for real combustion devices. The key step in such procedure is the *comparison* (assessment) of model performances with respect to experimental data. Experimental data are typically found in *supplementary materials* attached to scientific papers as CSV, ASCII or Excel files. While this good reporting practice is almost always the case in recent publications, older studies did not include such details. In this case, semi-automated or automated extraction of experimental data and boundary conditions from tables, figures and text reported in the manuscript is necessary.

From a data perspective, an experiment consists of a set of *conditions*, which describe the experimental setting, and a set of *output variables*. The conditions can be categorical values (e.g., the reactor type), constants (e.g., the initial mixture or the initial pressure of the experiment) or, less frequently, data series. The conditions are needed to replicate an experiment or simulate it with a model. Outputs variables are series of (x,y) values and can be represented as discrete curves. An example of an experiment description

Boundary conditions:

- *Reactor type*: shock tube
- *Pressure*: 4 atm
- *Initial composition*:
  - H2O = 0.25 mole fraction
  - N2 = 0.5036 mole fraction
  - O2 0.1339 mole fraction
  - H2 0.1125 mole fraction

Output variables:

- *Ignition delay time* as a function of the *temperature* in the 1051 – 1272 K range

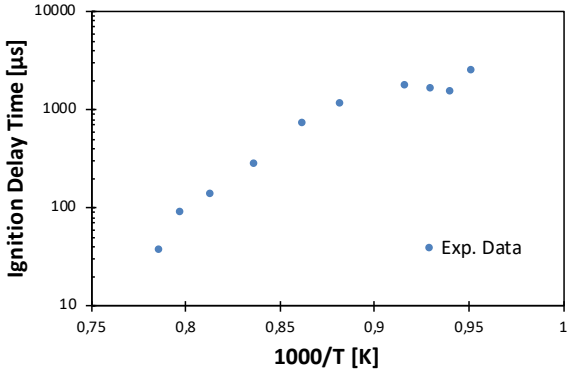

Fig. 2. On the left, boundary conditions and output variables of the experiment `10.24388x10000021_x`, extracted from [13] and available in the ReSpecTh repository [5]. On the right, the output variable *ignition delay time* is plotted with respect to the *inverse of the temperature* in a logarithmic scale, as generally accepted practice in the domain.

provided by the ReSpecTh repository [5] is shown in Figure 2, highlighting the extracted boundary conditions and output variables, which are also plotted. In ReSpecTh, the experiment description — originally in the XML format — includes also other data not shown in the Figure, including a unique DOI (`10.24388x10000021_x` in this case) and information about the source paper from which data have been manually extracted ([13] in this case, specifying also "Fig. 8., full square" as the source Figure). Supplementary materials of scientific papers, when available, include the same kind of information in Excel or CSV files.

A model's assessment is typically performed by using plots in which experimental data and curves from model simulations are plotted together. Researchers express their subjective evaluation based on their experience and knowledge, often neglecting experimental and model uncertainties. This standard model validation procedure was recently highlighted as a "poorly posed" problem [14]. Analysis tools (e.g., sensitivity analysis) allow the highlighting of relevant model parameters and drive their refinement by means of more accurate estimation methods.

An example of a plot which compares the output variables of the experiment detailed in Figure 2 with the same output variables as predicted by some models — given the experimental boundary conditions — is shown in Figure 3. In general, the concept of "best model" is not easily definable [9]. Historically, the evaluation of model performances has been performed by means of a rather subjective interpretation of the degree of agreement with experimental data as visualized on 2D (x,y) plots. Within the combustion kinetics domain, Turanyi and co-workers reported the sum of squared error based methods to evaluate the performances of different kinetic models [15, 16]. New techniques were recently proposed in order to automate a model's performance assessment with respect to some experimental data, deriving objective and quantitative measures and overcoming manual graphical comparisons. The Curve Matching algorithm [9] allows an objective, quantitative and automatic evaluation of the model's capability to predict an experiment's variables of interest, extending the most common, but sometimes misleading, sum of squared error based approach [17].

If the model provides satisfactory agreement, subsequent steps of optimization and reduction [18] make the model suitable for large scale computations of interest for industry (i.e. reactive computational fluid dynamics). Indeed, the developed model is not manageable given its size (number of chemical species and reactions), and model reduction, provided a desired degree of agreement with the original detailed model, allows to obtain a smaller model that is suitable for applications. On the contrary, if the

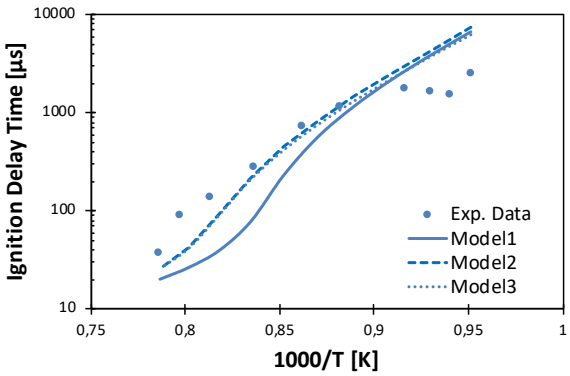

Fig. 3. The output variable of the experiment in Figure 2 is shown against some models which approximate it.

model shows deviations outside of experimental uncertainties, relevant pathways that can be identified by means of analysis tools and model parameters are further refined with better estimates. Indeed, recent developments coupling high performance computing and theoretical chemistry allow the automatic generation of highly accurate parameters [19, 20].

The efficient integration of the above tools and the process shown in Figure 1 in a fully automatized system has been recognized as one of today's challenges in kinetic modelling [20]. Exponential growth in the volume and complexity of scientific information in the combustion community (experimental data, models, theoretical investigations, etc.) and improved accuracy of experimental techniques can be beneficial, at best, only if coupled with efficient tools for acquiring, storing and analyzing such information, and therefore allowing real advances in knowledge. In this paper, we focus on an integrated system mainly targeting the *validation* phase (dotted rectangle in Figure 1).

Several initiatives to enable the effective and structured collection of experimental data for combustion science are described in the literature. Starting from the pioneering work of M. Frenklach and co-workers to develop the PrIMe database [8], which is still under continuous update, the ReSpecTh repository [5] largely improved and extended it by means of a more flexible, detailed and user-friendly data structure. At present ReSpecTh has collected more than 1,000 experiments into XML files, which correspond to more than 87,000 data points, and the extent of this collection is expected to increase in years to come. CloudFlame (KAUST) [6] provides an open database for experimental data in a standard CSV format, together with a cloud infrastructure for running simulations based on stored models and data. To improve human readability of structured data, ChemKED [7] has introduced a human and machine readable data standard for chemical kinetic experiments based on YAML, together with a set of validation and conversion tools. The COST Action CM 1404 (SMARTCATs[4]) established a task force of scientists to define standards for data collection, allowing easy and effective coupling with the above systems.

Over and above the reference repositories mentioned, one should consider a large amount of experimental information stored in less structured formats in many institutional servers belonging to experimental or modelling groups working in the field of combustion. As an example, the CRECK repository is the result of data collection from over ~ 40 years of research efforts in modelling combustion kinetic and thermal processes. While data management has relied on manual extraction and organization into unstructured spreadsheets or text files for a long time, a relational database with a local interface for querying and exporting data was recently developed [10]. However, the limitations of that software and

[4] http://www.smartcats.eu/

new use cases which go towards the direction of a system that not only stores experimental data, but leverages them to support and enhance the entire model development process, led to the design of a new infrastructure which includes not only an experimental database, but also integrated data curation and analysis tools. Requirements for the new infrastructure were investigated starting with [11]. This paper moves towards the same direction and describes, in addition to updated and more complete use cases and requirements, a new framework prototype with its designed architecture.

As discussed at the end of the following section, the need for continuous model validation based on new experiments and use cases arising from this domain are certainly shared by other scientific fields. This makes this development scenario, and its challenges in terms of a supporting infrastructure, a general scenario which is also valid in other domains with similar characteristics.

3. Related work

This section presents a brief overview of related works available in the literature. As the proposed framework includes challenges in data management, data quality and cleaning, each of them is briefly outlined below, before describing other systems with similar goals and other related work which motivated this system. Other papers will be cited in the following sections when discussing requirements on specific issues.

All requirements analyzed in this paper strongly rely on *data management* efforts. Indeed, it has been highlighted in the literature how value creation is not driven by data itself, but by the whole data management process [21], and that it is enabled by the management of both internal and external data. Data reuse, in particular, requires specific management techniques.

The goals of this work stem from the increasing availability of open and structured scientific data and the resulting needs of new ways to effectively leverage and reuse them. In general, the concept of "open data" is not easily definable and baseline conditions are "fewest restrictions" and "lowest possible costs" [22]. It has been noted how data sharing practices have been investigated more than data reuse [4]. Data reuse is not to be limited to reproducing research, which is an example of independent reuse (that is, reuse of an individual dataset) but should also include *data integration* "to make comparisons, build new models or explore new questions altogether" [4]. This is the direction of the present work, which focuses on reuse as a means to extract new knowledge not available by analyzing individual instances. To reach this goal, it has been highlighted how standards and formats for data release influence reuse opportunities. Standards are a prerequisite to understand and integrate datasets, but even with these in place, differences in parameters, instrumentation and other factors make data integration non-trivial [23]. Moreover, semi-structured data formats could add ambiguities to standardized datasets.

Challenges related to data reuse and value creation are discussed in [24]. Finding useful datasets is a challenge in itself, since this precedes their interpretation. This requires adequate metadata and often some kinds of standards, but "adequate" has a different meaning when the goal changes, and often a publication includes only the bare minimum required for its specific goal. There are also format issues, which sometimes can be simply a question of performing a schema mapping, but often require resolving more substantial mismatches for an effective integration.

Besides domain-specific repositories, described in Section 2, several general purpose scientific repositories have gained importance in the last years to efficiently share research outputs, including exper-

imental data. Notably, examples are Figshare[5], Zenodo[6], myExperiment [7] and Gigantum [8]. On top of these, tools to efficiently search the vast amount of published data, such as Google Dataset Search[9], and to efficiently reproduce scientific code, such as Code Ocean[10] have been made available. This work investigates ways to aggregate and analyze published experimental data for a different goal: supporting scientific model development. Collecting and storing experimental data is a means rather than an end for the proposed framework and the goal is to leverage existing efforts in data sharing as much as possible. Overcoming challenges like experimental data interpretation and integration, discussed in this paper, should pave the way for large-scale comparisons on published experimental data that could lead, for example, to *indirectly* detecting inconsistent results, rather than directly reproducing an experiment.

Two fundamental aspects of the proposed framework are data quality management and data cleaning, which are strongly related to the *veracity* of big data. Their importance comes from the reuse of data which can be noisy, outdated, misleading and, in general, unreliable. Indeed, incorrect information is often found in shared data [23] and unreliable experimental data have been identified in the chemical kinetics domain even quite recently [25]. Quality issues could be introduced even later in the processing pipeline, for example when experimental data are imported into a repository or converted into a specific format, given also the constraints of the related standard. Independently from the cause, it is important to assess data quality before using data in order to get meaningful results.

Data quality management techniques used in traditional databases are not enough to handle a big data scenario with heterogeneous sources, which instead requires an "adaptive approach able to trigger the suitable quality assessment methods on the basis of the data type and context in which data have to be used" [26]. Among other challenges, this requires *context-dependent* quality assessment, which takes into account, for example, that a large number of sources can instill confidence in the dataâĂŹs reliability, and *multi-granularity* assessment, to evaluate data quality at various aggregation levels [27]. Data cleaning techniques enable the detection and repair of data errors by identifying integrity constraints, duplicates, functional dependencies, and so on. These activities can be automatic or human-guided, and error detection techniques can be applied on the original database or later on in the data processing pipeline [28]. In this paper, the discovery of errors after some processing and integration is of particular importance given the use of large-scale cross comparisons and the integration of external sources. Such "delayed" cleaning has been analyzed and formally described in the literature [29]. Data integration, and in particular the analytical querying of an integrated set of data sources, is normally addressed through data warehousing (DW) and online analytical processing (OLAP) systems. However, in a traditional OLAP system all data sources should be known and described in advance, as the rules used to perform their integration, and external data sources (i.e., sources which evolve outside the system) are not taken into account. Moreover, traditional OLAP systems typically do not deal with semi-structured data and external ontologies. Such problems arise in this paper's context, given the need of dealing with multiple, variable and external sources of experimental data (scientific repositories) and external ontologies (domain knowledge which should be managed externally by domain experts) and to obtaining integrated results. Semantic-aware *exploratory OLAP*s have been proposed to face these limitations allowing the exploration "of new data sources, of new ways of structuring data, of new ways of putting data together,

[5] http://figshare.com
[6] https://zenodo.org/
[7] https://www.myexperiment.org/
[8] https://gigantum.com/about
[9] https://toolbox.google.com/datasetsearch
[10] https://codeocean.com/about

of new ways of querying data" [30]. Such solutions provide several benefits with respect to data extensibility and dynamicity and their "open-world" assumption allows analytical querying of new external data.

The framework investigated in this paper shares some features with scientific workflow management systems (WMSs). Workflows are "a set of interrelated computational and data-handling tasks designed to achieve a specific goal" [31] and a WMS aids in the automation of those operations by managing their execution and information exchange. Scientific workflows are typically *data-intensive* workflows and scientific WMSs have been proven crucial in data-driven science [32]. A notable example is Taverna[11], an open source and domain-independent WMS which integrates, among other things, specific web services for chemistry-related workflows. In this paper, we focus on use cases, requirements and challenges coming from the reuse and large-scale analysis of available scientific data to extract new insights, in particular, with respect to scientific model development. Unlike WMSs, we do not directly address the development and management of new experiments ("in silico" experimentation) or models, under the assumption that these activities occur externally to the framework. Workflows can become complex and resource intensive, therefore WMSs face problems such as scheduling and resource allocation in distributed environments [31]. A workflow consists of several computational tasks linked by data dependencies and their management by WMSs involves challenges such as resource provisioning, provenance management, performance variation, parallelism and fault tolerance [33, 34]. Therefore, our work is complementary to a scientific WMS, since the analysis tasks identified could be managed efficiently as scientific workflows in a large-scale and multi-user scenario.

Scientific data are typically attached to scientific papers or manually extracted from them successively. Therefore, scholarly data repositories, which typically contain information about authors, citations, figures, tables, etc., are often also a source of experimental results [35, 36]. Moreover, being able to analyze the sources of experimental datasets, as well as data themselves, may lead to new analysis opportunities. The management of big scholarly data has been surveyed in [35] and currently there are initiatives to publish open bibliographic citation information as linked data, such as Open Citations [37]. The feasibility of tools to explore such data has been demonstrated [38, 39], facing challenges related to statistical analysis, semantic exploration and visual analytics. Scholarly data analysis is often based on knowledge graph analysis.

The usage of graph structures as conceptual-modeling allows the storing and querying of data based on the relationships among objects, which is necessary to model inter-dependencies among entities and for data exploration, also through faceted search [40]. Graphs can also be exploited for mining activities [40] and to model varying precision and accuracy. Graph modeling can be accomplished through a graph database [41] or via mappings that enable graph reasoning over traditional relational databases [42, 43].

Recent research results on the generation of knowledge from large-scale analysis of experimental data in the combustion kinetic domain have been reported in [25]. Hansen and co-workers presented a collection of 55 experimental datasets in premixed laminar flames extracted from previous publications, showing how their analysis allows the extraction of fuel-specific rules that can be useful to identify inconsistent data. Along the same lines, [44] also collected and reviewed a large set of experimental data (60 datasets) on rich laminar premixed flames of hydrocarbon fuels reported in recent years, and analyzed them by means of the available kinetic models to describe the formation of polycyclic aromatic hydrocarbons. The Curve Matching approach [9] was applied in the latter case to evaluate model performances. This allowed the identification of inconsistencies not only between single models and

[11] https://taverna.incubator.apache.org/

experiments, but also between different models often describing the same phenomena in significantly different ways, yet still obtaining a comparable degree of agreement. From these recent literature examples, it is clear how a fully integrated framework with large-scale capabilities, as that described in this work, could be beneficial to both model development and experimental evaluation.

The needs which emerge from the scenario described in Section 2, like the continuous validation of models based on new experimental data, are very general and shared by other domains and contexts. Therefore, the solutions discussed in this paper have a general validity beyond the domain of combustion kinetics. Indeed, recent literature has highlighted "cases where datasets are reused in combination with other data, whether to make comparisons, build new models, or explore new questions altogether" [4], in fields such as computational biology [45], gene expression analysis [46], biomedical research [47], cell biology [48] or predictive structural materials science [49]. All these fields are characterized by the mathematical, physical and computational modeling of complex systems and behaviors, and the validation of such models against experimental data which typically come from research articles. Indeed, similar challenges have been discussed recently in these fields, particularly those related to data integration, interpretation and large-scale analysis [45–49], highlighting several shared open issues. The framework investigated in this paper has been currently tested and optimized only for the scenario described in Section 2, but most of the design choices were taken trying to generalize its applicability.

Large-scale validation of models with respect to experimental data is an emerging topic and other integrated systems have been proposed very recently in different domains. PRISMS [49] is an integrated framework for accelerating predictive structural materials science. It includes computational modules to predict microstructural evolution and the mechanical behavior of structural metals and a set of integrated scientific "use cases" where they are linked to experiments. It also includes a collaborative repository to archive and disseminate experiments and computational models. CaRMeN [50] is a tool that automates the workflow for comparing chemical kinetic models and experiments for catalytic process applications, overcoming the issues of a manual, time-consuming and error-prone validation. It contains reactor solvers for different kinds of reactors, different models to simulate the catalytic partial oxidation of methane and related experimental data for validation. Model performance assessment is based on parity plot diagrams and the graphical interface makes it user friendly. Our work shares several goals with these projects. However, our efforts are directed more towards the management of challenges coming from the use of heterogeneous and potentially noisy data sources and their semantic interpretation to enable new analysis directions, thus obtaining requirements which are not domain specific.

4. Towards an integrated framework

Model development and automatic validation through published experimental data has several facets. In order to analyze the directions of an integrated system against this goal, the main use cases are analyzed in Subsection 4.1. Moreover, a first extended prototype was developed and presented in Subsection 4.2. This will be the basis for the formulation of new requirements and an architecture for the complete framework.

4.1. Use cases

4.1.1. Experiment simulation

An experiment that is stored in the framework can be automatically simulated by a stored model. Simulating an experiment means executing the model at experimental boundary conditions (initial tem-

perature, pressure, fuel/oxidizer mixture composition, residence time, reactor, reactor type, and so on), so that the simulation environment is as much as possible similar to the actual experiment, and comparing the results.

Simulation requires *simulation software*, used as an external service, such as OpenSMOKE++ [12]. The simulation software takes the target model and the experimental boundary conditions as input and returns the output variables as predicted by the model.

Therefore, automatically testing a model on an experiment requires:

(1) Building a *simulation input* for the simulator, which specifies i) experimental boundary conditions and ii) the target model;
(2) Executing the simulator with the simulation input;
(3) Comparing the model and the experiment outputs.

Steps 1 and 3 involve several sub-steps such as validation, transformations and post-processing.

The final comparison results in a *similarity score* for each experiment/model output which expresses the extent of the match, that is, the model's ability to correctly approximate the experimental results. The design of such matching techniques is outside the scope of this paper. Currently, the framework uses the Curve Matching algorithm [9] for this scope as an external service, but different matching algorithms could be supported.

4.1.2. Managing new experiments

As already discussed in Section 2, new experiments could be imported from different sources. Scientific and scholarly repositories represent a valuable resource for data already extracted from publications and stored in semi-structured formats (e.g., XML, YAML, or other custom formats). In other cases, new experiments could be directly extracted and inserted manually by domain experts from supplementary material attached to publications or from data and plots included in such papers.

In this phase, it is necessary to guarantee consistency, uniqueness and quality for all acquired information by avoiding missing data and acquisition-related errors. However, validation brings some challenges. Indeed, if on the one hand validation routines and quality checks are necessary to avoid input errors and partial information, on the other hand, an experiment type is not characterized by a fixed set of fields. Even if scientific experiments follow well codified and standardized procedures, as their description in scientific literature, many variations could occur. For example, some values could be described at a different aggregation level (e.g. indicating an average temperature instead of a temperature which changes slightly during the experiment), some papers could report more experimental datasets than others, some values could be missing in a paper because there exists a "standard" or "default" value (e.g. the atmospheric pressure) and there could be different sets of values which bring the same information (e.g. a time and a distance rather than a flow velocity). In general, there is no pre-defined level of resolution to describe a scientific experiment, but a sufficient set of conditions can be found that describe an experiment well enough to be understandable by a domain expert and reproducible with enough precision in a real setting or, as in our case, an automatic simulation.

The knowledge required to interpret and understand an experiment, and the set of conditions needed to effectively simulate it, are complex and domain-specific. Moreover, they could evolve as the model evolves or some assumptions change. Therefore, input validation can detect inconsistencies (e.g. wrong units for a certain variable), but the actual interpretation of the experiments should be dynamic and postponed. Similarly, in the acquisition phase, data should never be modified and an experiment should simply be kept in the database without changes, even if the knowledge to automatically understand it

is not yet fully available. There are two main reasons why experimental data should not be modified in their original description:

- Assuming an experiment has no input-related errors, it is a "source of truth" by itself and the knowledge to automatically understand and simulate it could be available successively.
- This allows the definition of an external "domain knowledge" to dynamically interpret, change (aggregate, convert) and use experimental data for simulation and analysis tasks.

Note that this kind of validation, including dynamic model-driven validation, only concerns the completeness, reproducibility, coherency and understandability of an experiment in terms of its "schem", that is, the set of conditions and variables included, but not the reliability of its data. The latter requires the experiment's comparison to other experiments/simulations and being able to correctly interpret it is a pre-requisite.

4.1.3. Exploratory and cross analysis

Experiments and models in the framework can be searched and filtered by a generic set of fields. This makes it possible to only select those experiments/models which satisfy some conditions and to execute cross analyses on them. This enables the discovery of behaviors and patterns which cannot be derived from the analysis of a single instance, and allows the handling of volumes of data largely beyond those manageable manually.

Two main tasks in this context are *model validation* and *experiment validation*, which aim to validate a model or an experiment's quality through large-scale comparisons and aggregated values, but generic *data analysis* processing should also be supported. All these tasks build on stored experiments and models and on the framework's automatic simulation capabilities, but also on domain knowledge which the framework integrates to perform semantic-aware analysis and enhance results.

These analysis directions are detailed as follows:

- *Model validation* allows the assessment of a model's *global* performance in specific conditions. Once some conditions are expressed through a query, the model can be evaluated with respect to all experiments satisfying the query and, optionally, with respect to other models for the same query set. Given the fact that each model-experiment pair brings an *index* which expresses the model's performance on the experiment (i.e. the extent of their match), such large-scale validation requires *data aggregation* to provide one or a few common indices for each model, e.g. a common index for each output variable, thus producing common performance indicators for the model.
- *Experiment validation* allows assessing the reliability of experimental data through large-scale cross-comparisons. Indeed, it was recently highlighted that correlations which exist among similar experiments allow the identification of inconsistent data [25] and the same approach can be extended to compare multiple experiments to model outputs.
- Finally, generic *data analysis* functions need to be supported. In this regard, the framework needs to be extensible and the functions to be assembled in *workflows* to further analyze and aggregate simulation results. Analysis examples in this respect are: *outlier detection* to discover experiments for which a model is less accurate, *clustering* to discover classes of experiments that lead to inconsistent results and *correlations* of model results with various (meta)data, such as *scholarly data* to examine the impact of experiments published by a certain author or journal.

All these analyses should be built around the concept of *exploratory computing* [51]. This involves an iterative and multi-step process where results are summarized and visualized, thus iteratively refining the initial query.

4.1.4. Managing changes in models

Whenever a model is modified, it could simply be considered as a totally new and independent instance, and validated against the whole set of stored experiments computing its validation indices. However, such an approach has at least two major shortcomings: lack of development tracking and required computational resources.

Development tracking. In almost all cases, a new model is a modified (potentially improved) version of an existing model along the development process already shown in Figure 1. This means that improvements should be expressed not only in absolute terms, but also, and more importantly, in relative terms with respect to the starting point. This reflects the fact that a change in a model has the primary goal to improve its performance. Therefore, by considering a new model as a variation of an existing one helps track relative changes. By doing so, the continuous "test and refinement" cycle leads to a *sequence of models*. For each pair of models of this sequence, it should be possible to derive both the relative change in the model and the relative change in the output performance, allowing, ultimately, to link model changes to performance variations. In some cases, the user could have more of an exploratory intent. There could be the need of parametrizing an initial model and simultaneously testing different combinations to analyze the impact of one or more model hyper-parameter(s) on the resulting performance. In any case, a modified model should be tracked as a variation of the starting one, and, at the same time, could be "parallel" to the other variations.

Computational resources. As mentioned above, testing a model on an experiment requires building a "simulation input" for the model, executing it and comparing outputs from the model with experimental outputs. In the majority of cases, the actual model execution is the most demanding task as regards time and resources.However, independently from the actual time/resources needed to simulate *a single* model-experiment pair, bottlenecks could also arise following the development cycle and from continuously re-testing modified model versions on the whole set of experimental data. Considering a new model as a variation of an existing one helps in this, because a change in the model could only affect *a portion* of its behavior, which, in turn, could only affect a subset of the whole experimental dataset, thus significantly reducing the number of executions needed. Moreover, from a more technical point of view, the re-execution of a slightly changed model could benefit from intermediate results already obtained by executing the starting model, thus reducing the time and resources needed.

Therefore, it becomes crucial to follow model development and to keep track of versions and derivations. Once a model changes, it should be possible to re-validate only the portion of experiments affected by the change.

4.2. Extended prototype

The new prototype has been developed to better analyze emerging use cases and requirements, and, therefore, we followed an *agile* development approach. For this reason, it already includes a set of fully working features, some partially developed ones, and a set of demo features developed to better refine requirements and to design the architecture. Much effort was spent in developing the core integrated infrastructure and in the abstraction and integration layers which will allow further development and this prototype's evolution in a full framework. The source code of the prototype is available[12]. The full framework's architecture, which was also designed and refined starting from this prototype and includes parts that are not yet developed, is described in Section 6.

[12] https://github.com/sciexpem/sciexpem

Currently, the prototype includes the following (fully or partially developed) main features:

- Batch import of experiments into ReSpecTh [5] and ChemKED [7] formats[13]. Folders of such experiments can be manually imported, but automatic import/update directly from these repositories is currently not supported.
- Manual input of experiments extracted by domain experts through a mixture of interactive input fields and human-readable `Excel` and `CSV` files.
- An interface to add kinetic models to the system. A model is uploaded as a folder and associated to a name and a version.
- An *operational database* which stores experimental data, logical paths to models and the results of simulations and analysis performed by external tools. All these data have been described through a generic meta-model with generic fields to facilitate data integration and manage different sources. For example, experiments are described in terms of "boundary conditions" and "output variables", and the interpretation of the actual values is dynamically delayed and supported by an external domain ontology.
- Automatic recognition, validation and cleaning of experiments based on an external ontology. Currently, a simple ontology has been defined using an Excel human readable file and populated by domain experts, as discussed below.
- An interface to browse experimental data and visualize their properties and output variables, also through interactive graphs with options to compare, highlight, zoom in, etc. This interface is general enough to include different experiment types, as long as they are described in the ontology.
- Automatic creation of *simulation inputs* starting from (one or more) experiment(s) and a set of models to be simulated, and the parsing and storing of simulation outputs. This allows the automaticalal simulation of every interpretable[14] experiment. Currently, only the OpenSMOKE++ simulator is supported.
- Search and filter experiments based on their properties, input species, and a generic boolean condition.
- External tools supported with a modular design. Given a set of experiments obtained as a result of a search query, it is possible to run a generic analysis on them which can be defined as a standalone function receiving each experiment with associated data as an input and new data which is stored and associated to each experiment or to the entire set as an output. This "search & execute" pipeline is the prototype's core. Function chaining is not supported yet.
- Automatically executes the Curve Matching algorithm on a set of experiments and models to evaluate the models' performance on each experiment. An interface to the Curve Matching algorithm [9] has been implemented as a module.
- Compute a global score for each model on a set of experiments, starting from the indices defined for each model/experiment pair by the Curve Matching. This function has been implemented as a module.
- An interface to browse simulation outputs and analysis results. Once an action has been executed on a query set, it is possible to browse its results both in a *global view*, which shows set-level results, and in a *detailed view*, which shows experiment-level results. These views are currently able to

[13] The ChemKED format is supported through the PyKED utility (https://github.com/pr-omethe-us/PyKED) which allow to convert a ChemKED file in an equivalent ReSpecTh file.

[14] An experiment is interpretable by the framework if its definition has been added to the domain knowledge. More information about this is given below.

```
{
  - data: [
      - {
            model: "polimi1412",
            average_index: 0.8634703,
            average_error: 0.0027774,
            CO2_index: "0.8057320",
            CO2_error: "0.0041761",
            CO_index: "0.8559380",
            CO_error: "0.0015967",
            C2H5OH_index: "0.9287410",
            C2H5OH_error: "0.0025594"
        },
      - {
            model: "polimi1800",
            average_index: 0.877474,
            average_error: 0.0032408,
            CO2_index: "0.8177820",
            CO2_error: "0.0054314",
            CO_index: "0.8670340",
            CO_error: "0.0013645",
            C2H5OH_index: "0.9476060",
            C2H5OH_error: "0.0029266"
        },
      - {
            model: "polimi1809",
            average_index: 0.8777867,
            average_error: 0.0034936,
            CO2_index: "0.8198380",
            CO2_error: "0.0062727",
            CO_index: "0.8659550",
            CO_error: "0.0012850",
            C2H5OH_index: "0.9475670",
            C2H5OH_error: "0.0029230"
        }
    ],
  - names: [
        "CO",
        "CO2",
        "C2H5OH"
    ]
}
```

```
[
  - {
      id: 1902,
      reactor: "stirred reactor",
      experiment_type: "jet stirred reactor measurement",
      fileDOI: "10.24388/x00003001",
    - common_properties: [
      - {
            id: 2337,
            name: "pressure",
            units: "atm",
            value: "1.0000000000",
            sourcetype: null,
            experiment: 1902
        },
      - {
            id: 2338,
            name: "volume",
            units: "cm3",
            value: "30.0000000000",
            sourcetype: null,
            experiment: 1902
        },
      - {
            id: 2339,
            name: "residence time",
            units: "s",
            value: "0.0700000000",
            sourcetype: null,
            experiment: 1902
        }
    ],
    - initial_species: [
      - {
            id: 3665,
            name: "N2",
            units: "mole fraction",
            amount: "0.9740000000",
            cas: null,
            experiment: 1902
        },
      - {
            id: 3664,
            name: "O2",
            units: "mole fraction",
            amount: "0.0240000000",
            cas: null,
            experiment: 1902
```


Fig. 4. Examples of JSON responses from the REST interface. On the left, validation results are returned for a set of requested models and experiments. On the right, the list of experiments satisfying a certain query is returned and each experiment includes its metadata, boundary conditions and actual data (this response is not complete in the figure).

show model simulation outputs and Curve Matching validation indices, including global scores and derived graphs, but are not yet general enough to show the results of third party modules.

These features already cover part of the use cases described in Subsection 4.1. The main features currently missing from the prototype include those related to automatically "make sense" of the results, e.g. automatically understanding the reason why a certain model does not behave correctly on some experiments, also through the dynamic integration of other data sources, the dynamic acquisition and exploration of new experiments and the integration with external systems (e.g. WMSs and model development tools). The prototype helped to highlight challenges arising in the development of use cases, which led to the new requirements discussed in Section 5 and the architecture outlined in Section 6. The latter can be considered as this prototype's future development.

The client application was developed as a web application to minimize client-side requirements. The server offers all its services through a REST API in JSON format, allowing integration with other applications and the possibility to develop alternative clients. Examples of JSON responses are shown in Figure 4. User authentication is supported.

The prototype was implemented with PostgreSQL[15] as the operational database, using its extensions to support array data and semi-structured XML and JSON data. The backend was written in Python,

[15] https://www.postgresql.org/

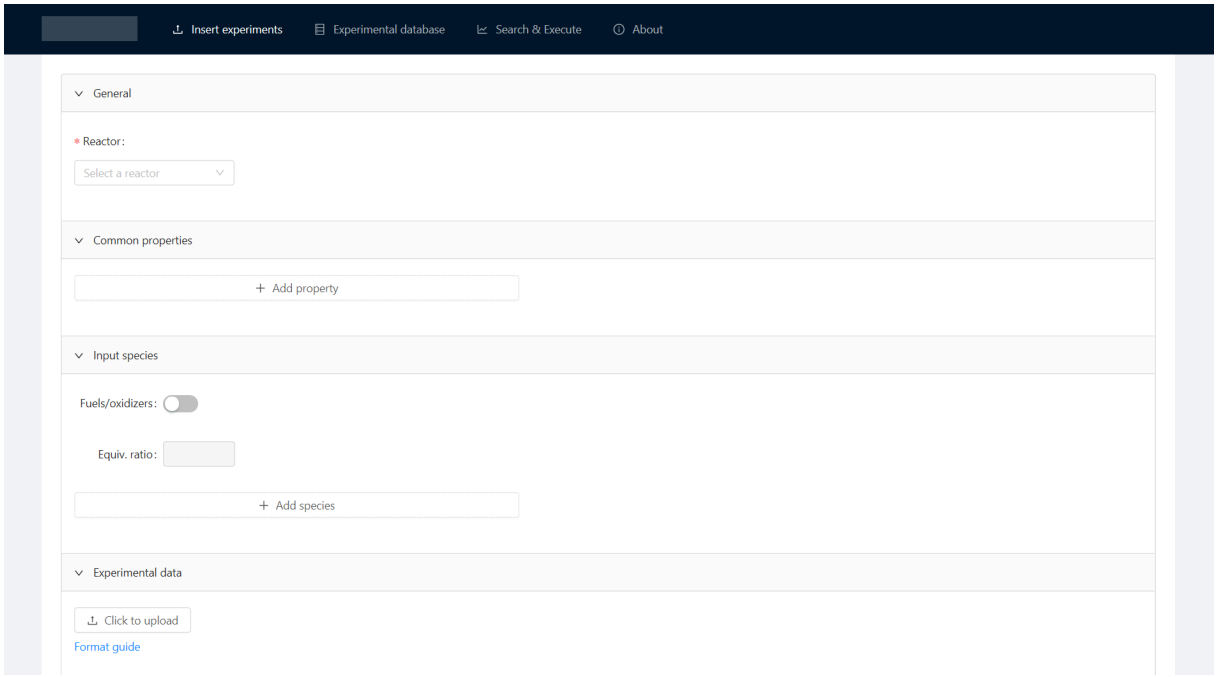

Fig. 5. Prototype Screenshot/1. Manual input of an experiment in the framework.

using the Django[16] framework as a web framework with the Django-REST[17] extension for the REST API. Python libraries used in the backend include: `numPy`, `sciPy`, and `pandas` for data elaboration, `pint` to manage physical quantities and conversions, `xlrd` and `elementtree` for data conversion. The frontend is based on the React[18] and Ant Design[19] frameworks and Plotly[20] was used as the graphing library.

At the moment, the prototype stores more than 1,200 experiments. They include those available in the ReSpecth respository[21] and manually uploaded experiments, including the collection published within [25]. The number of experiments stored is growing constantly.

In the architecture, the understanding of experimental data leverages an external ontology which is populated by domain experts and, in the current prototype, is built starting from an Excel human readable file. These files were designed in the structure and populated in the content for the occasion. They allow the definition of existing categories of experimental data and the properties, variables and conditions required to simulate each category, including optional fields. This information makes it possible to perform data cleaning, correctly build the simulation input for experiments and interpret their output variables. Externally defining this information in a human-readable format leads to more flexibility and gives domain experts the possibility of extending support to new experiments without changing the core system. Experiments can be stored without semantic constraints as long as they satisfy schema

[16] https://www.djangoproject.com/
[17] https://www.django-rest-framework.org
[18] https://reactjs.org/
[19] https://ant.design/
[20] https://github.com/plotly/plotly.js/
[21] http://respecth.hu/

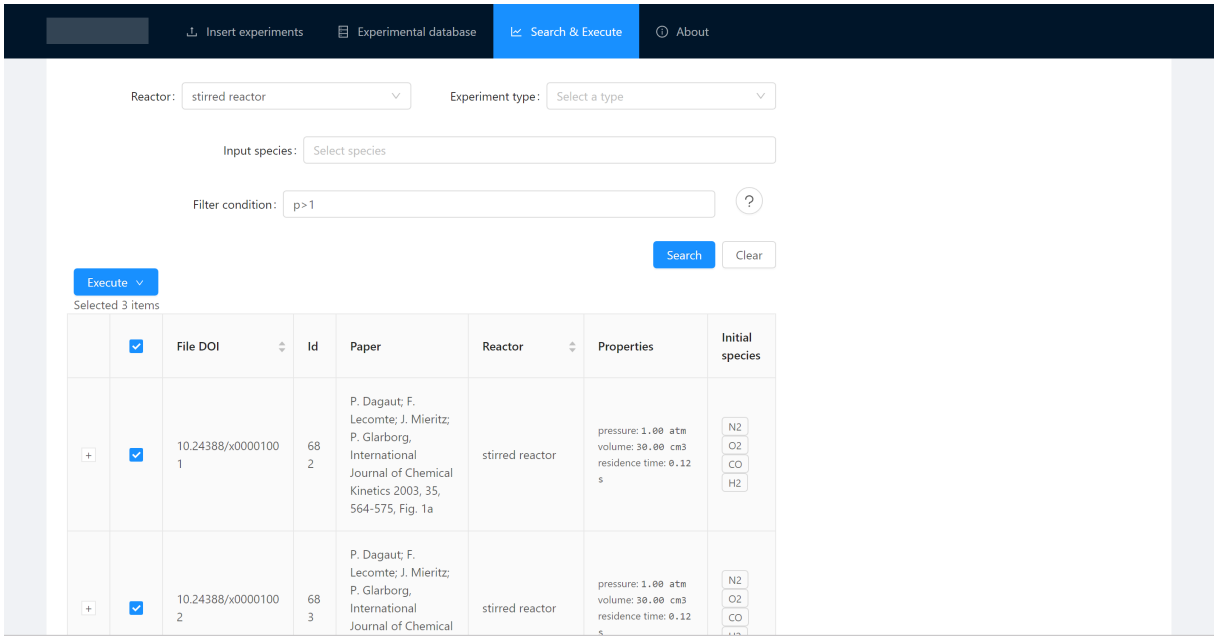

Fig. 6. Prototype Screenshot/1. Search experiments by complex filter conditions and select the resulting subset.

constraints, but their semantic interpretation, which happens dynamically based on the ontology, is then required for their usage (simulation and analysis) in the framework. This allows, for example, the storing of experiments not yet manageable by the system, which are automatically managed as soon as their information is added to the ontology. At the moment, about one third of the stored experiments are interpretable by the prototype, but this number is growing rapidly as new domain knowledge is added. The ontological information developed for the current prototype could be substituted by a more complete and complex domain ontology based on a more flexible format. However, such a complete and open ontology does not presently exist for the domain and should be generated as discussed in Subsection 6.1.

Some client-side prototype screenshots are shown below. Figure 5 illustrates the interactive interface to manually upload new experiments. Figure 6 shows the search form through which it is possible to filter experiments based on a generic query and to select them. Figure 7 illustrates the interface through which a list of models can be selected and then validated on the set of experiments previously filtered using Curve Matching with the OpenSMOKE++ simulator. Figure 8 shows this validation's global results consisting in an average *index* in the range $[0, 1]$ for each model and each output variable. From there, detailed results can be browsed, as shown in Figure 9. Detailed results include experiment/model curves and the validation index for each experiment, model and output variable.

5. New requirements

A new set of requirements have been formulated starting from the scenario presented in Section 2 and the use cases described together with the prototype in Section 4. The goal of these requirements is to enhance the efficiency and effectiveness of data management and to enable new exploration and analysis

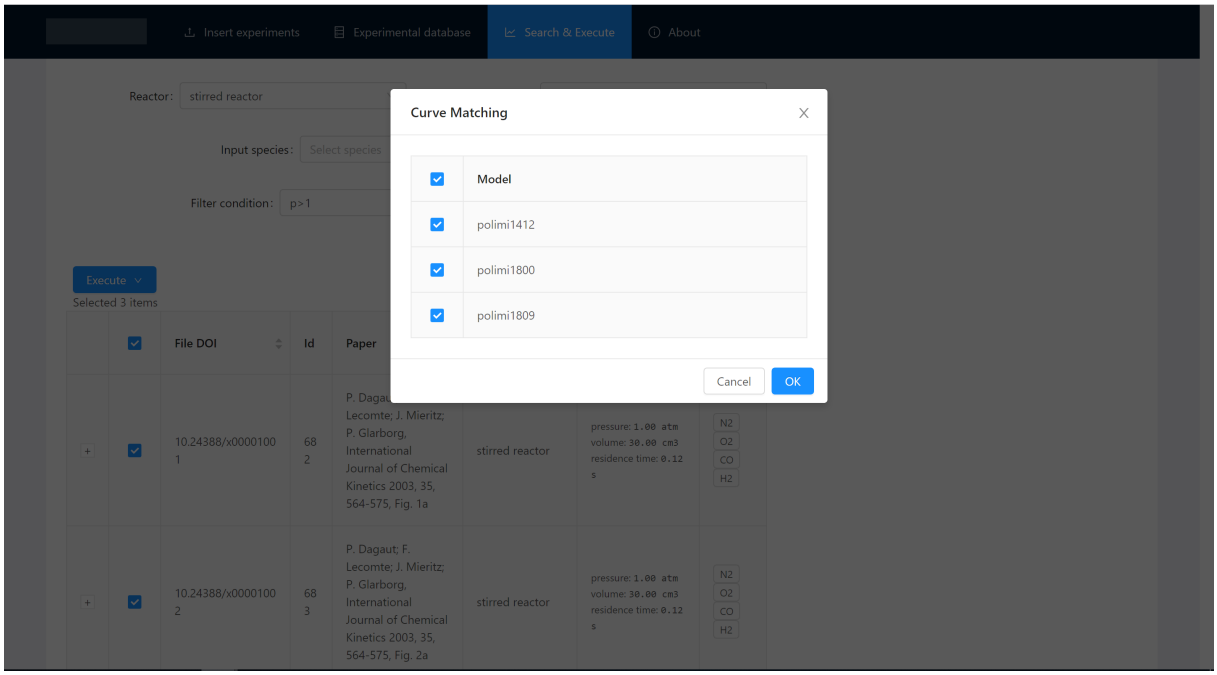

Fig. 7. Prototype Screenshot/2. Select the models to be validated and compared against the previously selected set of experiments.

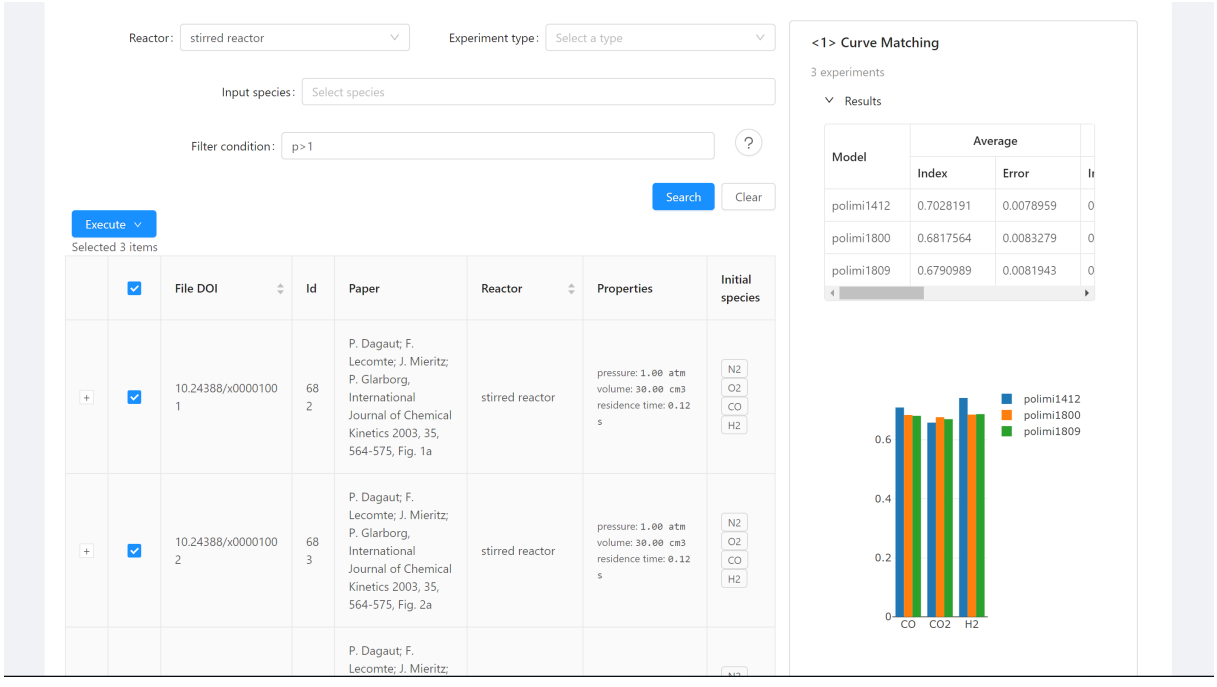

Fig. 8. Prototype Screenshot/3. Aggregated results are shown for all the experiments and models selected. An aggregated index in the range $[0, 1]$ is computed for each output variable and each model and is also shown as a bar chart.

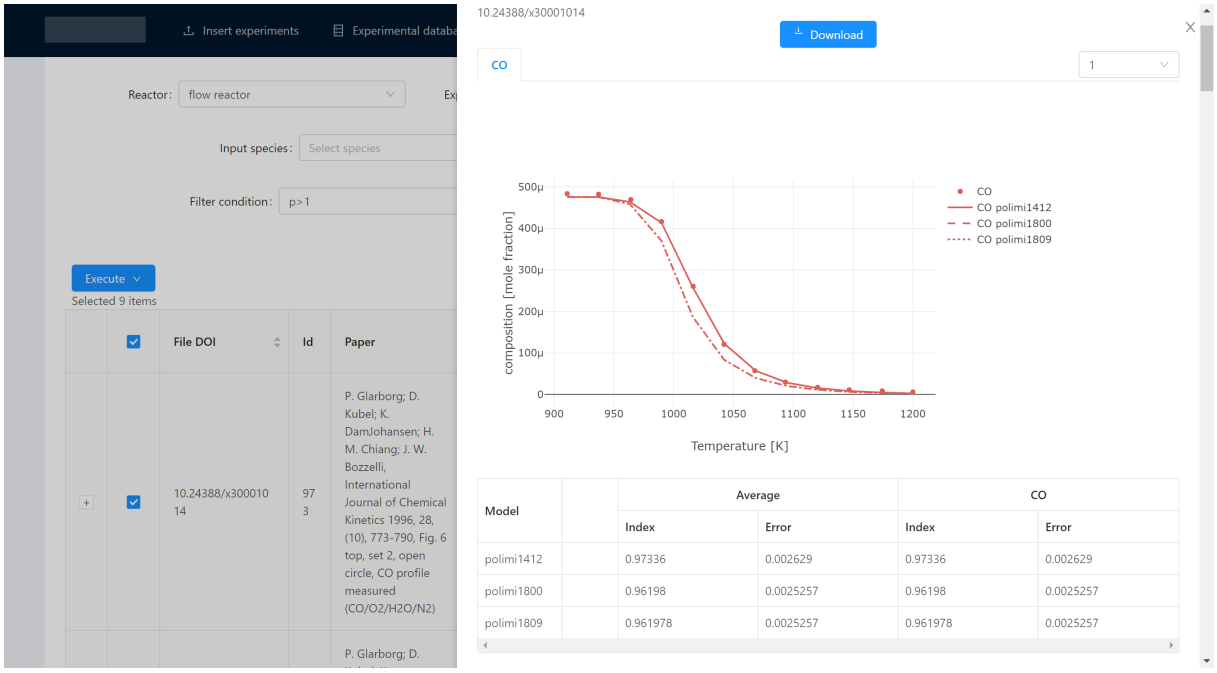

Fig. 9. Prototype Screenshot/4. Detailed results can be shown upon request; in this case the index in the range $[0, 1]$ is shown for each model/experiment comparison, together with a chart reporting raw experimental outputs (dots) and related simulated outputs for each model (lines).

directions in this context. They are presented and discussed below, and will also drive the design of the new architecture presented in Section 6.

5.1. Continuous multi-source integration and quality management

The need for continuous multi-source integration comes from the variability of the information sources, which could vary over time and cannot, therefore, be assumed a priori. This integration has many facets. First of all, the *format* and *structuredness* of data varies. A *semantic* integration is deemed necessary, since the same concept could be described differently in different instances, and this requires an ontology-based semantic layer and a dynamic interpretation of the information already stored. The information conveyed by the data is heterogeneous and can vary largely in terms of *accuracy*, *precision* and *coverage*. Continuous management of the already-acquired information is necessary in order to re-interpret data already stored according to new knowledge, which could also derive from different sources.

Such a dynamic environment impacts *information quality* (IQ) management: IQ — with each single dimension that defines it — evolves as new data and metadata are acquired or generated through processing. Stored objects do not only change over time, enriching their information, but are also characterized by explicit (*complex networks*) or implicit (*articulated objects*) interdependencies. For example, an experiment stored in the framework may have a certain quality which depends on its data, but further acquired "similar" experiments and their analysis could lead to the discovery of inconsistencies in the first experiment, thereby affecting its quality, without changes in the experimental data itself. IQ of such articulated objects is a function of the information quality of sub-objects and of other objects for which

a relationship exists. Indeed, besides managing the quality of raw data, which is a problem addressed in the literature, the focus is on introducing a way of managing the quality of complex information through their relationships. Data dependencies over these relationships are in general not simply additive and certain characteristics of an object (an experiment or a model) emerge only through large-scale comparisons.

Knowledge extraction and IQ management from complex relationships can benefit from the availability of domain ontologies as data sources and their dynamic integration in the analysis. At present, an open issue is represented by the lack of a complete domain ontology.

5.2. Continuous dynamic validation

The process of continuous validation entails the matching of already stored information with new information as it is acquired. This requirement mainly comes from the need of validating *models* and *experimental data*, against each other through curve matching techniques, when new experiments are acquired, or when a model evolves, as described in Section 4.

Validation is performed through *cross-comparisons* which require efficient means for *extracting* the right data, *comparing* them by taking into account the differences, deficiencies and uncertainties that could exist in their representations and *enriching* the objects (models and experimental data) with the results. To do this, multi-source integration is a prerequisite, especially to extract the right data. For example, to extract the experiments that need to be re-validated when a model evolves, it is necessary to assess the change in the model (model data), derive the set of experimental conditions affected (domain ontology) and extract the experiments which satisfy these conditions (experimental data).

Validation also requires investigating the *reason* for variations in the obtained result, since there are often many different situations that must be taken into consideration. For example, if some models fail to simulate a new experiment:

(1) The experiment could have some information which is missing in its original description, or missing information could derive from the partial acquisition of the experiment's information.
(2) The experiment could have been acquired with all the necessary information, but its interpretation by the framework is not complete and leads to wrong simulation constraints and misleading comparisons.
(3) The experiment could actually be imprecise/unreliable.
(4) The models could be unable to precisely simulate the condition described in the experiment because something is missing or imprecise in its design.

The consequences are very different: the first two cases should be addressed by modifying existing experimental or knowledge-related data in the framework, the third case requires labelling an experiment as "not reliable", while the fourth case should drive further testing and model refinement. More data can assist in the specific instance's identification.

5.3. Dynamic acquisition

Large-scale validation and analysis tasks benefit from a large amount of data, which include experimental data but also other complementary information, such as scholarly data, as an enabler of more complex analysis tasks. Indeed, data scarcity could severely impact most of the outlined analysis. In order to effectively enrich the framework's data repository, overcoming semi-manual data input, a requirement is therefore represented by the automatic and dynamic acquisition of new data.

This should be accomplished by a dynamic acquisition driven by the data already stored. The goal is to enhance the IQ of these data and improve data coverage by acquiring new information (for example, new experiments to benchmark existing models or scholarly information related to some already acquired experiments). This can be obtained by extending the concept of "focused crawling" [52]. The best predicate for querying new information from external sources, in general, changes over time and depends on a background knowledge and already stored information. Background knowledge is certainly composed of already acquired information, but also from the list of preceding queries and their results. Indeed, when acquiring data from unreliable sources it is not possible to make strong assumptions about them and an *exploratory approach* must be employed. Acquired data can drive the acquisition of new data in a *virtuous cycle*. For example, given some experimental data acquired from a certain source which does not include details about the original publication, the related scholarly data could be acquired from a different source, and this could lead to the acquisition of all the experiments published by the same author of the first source.

5.4. Data exploration

This requirement arise from the need for automatic or manual querying of information which is, in general, incomplete, heterogeneous or may not exist at all. Articulated (meta)data can provide support for interactive and evolving data exploration and *explorative computing* [51]. Specific techniques are needed to efficiently browse and *explore* the vast amount of stored data. These include summarization [53], result refinement, iterative exploration [54] and a *top-k* query processing environment [55], thus improving the returned "manageable" results.

6. Proposed architecture

The framework's designed architecture is outlined in this section. Starting from the previously-discussed requirements and the new prototype's development, this section will focus primarily on *architectural requirements* needed to support the use cases described in Subsection 4.1. A detailed architectural design, together with its complete development and testing, are out of the scope of this paper and represent ongoing work started with [11] and continued with this paper.

As previously illustrated in Subsection 4.2, some components and features have already been developed in the prototype. Beside these, the following architecture includes other components and features that must be considered as not yet having been developed. The development of existing features in the prototype has proved to be crucial to refine the following architecture.

Subsection 6.1 below describes the data model, while Subsection 6.2 gives an overview of the global architecture.

6.1. Data model

System requirements discussed in Section 5 translate into a set of data-related requirements:

- Supporting *continuous data integration* to combine data from different sources, including external sources, without assumptions on their number and type (structured and semi-structured) and allowing each individual source to evolve independently and to dynamically handle data and format conflicts.

- *Analytical querying* to support the various kinds of analysis, which are in general expressed at high level and inherently combine various sources of information at an aggregated level.
- Aiding the *exploration* and *discovery* of relevant data through user-defined ontologies and reasoning capabilities.

Most of these requirements can be addressed through *exploratory OLAP* systems supported by semantic technologies (see Section 3). Technological aspects of such systems are outside the scope of this paper, therefore, hereunder we focus on the discussion of which data sources need to be continuously integrated, queried, and discovered to support the requirements and analysis tasks described previously.

Several data sources can contribute to analysis tasks:

(1) The *operational database*, which stores experimental data, models (as logical paths, without internal information) and the results of simulations and analysis performed by external tools. This is a relational database continuously fed by the framework.
(2) Internal structure of *scientific models* to extract associated detailed information, perform comparisons and keep track of the development work-flow.
(3) *Domain ontologies*, which can support the integration, exploration and analysis of all the data and evolve independently and externally with respect to the framework. These are semi-structured linked data.
(4) *Scientific and scholarly repositories*, which can be the source of new experimental data and aid the integration, exploration and analysis tasks. Note that a repository could be a source for scientific data, scholarly data, or both.

Each of these sources is detailed below.

Operational database. The operational database is the base for simulations and analysis tasks. Its main goal is to uniquely identify each experiment, model, simulation and analysis outcome. This guarantees consistency and avoids the re-execution of the same model on the same experiment and allows re-using results from intermediate analysis. The operational database has been designed as an extended relational database to privilege performance and general enough to ensure the same schema for all stored data. For example, an experiment is described only in terms of reactor information, boundary conditions and output variables and no semantic constraints are put on conditions or variables names. Indeed, experimental data's semantic interpretation is dynamically delayed (that is, executed at *run-time* when an experiment is needed) and mediated by the domain knowledge (external ontologies), as discussed in Subsection 4.2. The operational database does not include data that is not strictly needed to simulate a given experiment with a certain model. Therefore, for an experiment, it stores its data (conditions and output variables) needed for the simulation, but not the related detailed scholarly metadata and, for a model, it stores only the model "logical path", necessary to execute it, but not its internal structure. All these characteristics allow for a compact and regular schema and reliance on the dynamic integration of external and potentially less structured data sources (like scholarly repositories) to perform more complex analysis, interpretations and validations.

Scientific model. A scientific model, besides being "executed" by the simulator, can also be seen as a data source. Indeed, its internal information can provide hints about its results and this makes it possible to link changes in its internal structure to changes in its performance. A combustion kinetic model describes the network of reactions, and the relative rate at which these reactions proceed, through which a reactant, or a mixture of reactants, is converted into products. While many disciplines dealing with kinetic modelling (e.g. catalytic processes) strongly rely on largely empirical and simplified models,

the relative simplicity of gas phase combustion is allowed to gain an extremely high level of detail thus resulting in complex kinetic models on the scale of hundreds of chemical species and thousands of elementary reaction steps, plus thermodynamic parameters and transport properties. Overall, the number of strongly interconnected model parameters can be on the scale of hundreds of thousands. From an operational point of view, a kinetic model is a structured file. It could be translated in the future into a graph data model, such as RDF, to be queried. Models evolve externally with respect to the framework and accessing their internal data could enable some analyses which are not feasible when considering them as black boxes, e.g., linking model internal changes to performance improvements.

Domain ontologies. Domain ontologies are needed to interpret experimental data and results as well as enabling their integration, exploration and analysis. For example, they describe which subsets of boundary conditions are semantically equivalent and can be converted from one to another, which boundary conditions should be associated with an experiment to be reproducible with the simulator, which are the output variables that need to be validated for each experiment type, and so on. An ontology is managed as an external data source because it can be imported from existing open data and it is managed outside the framework, even if purposely created for the framework when such information is not already available. As an available domain ontology does not presently exist, semi-automated generation techniques [56] or data mining techniques to automatically generate ontological relationships based on existing data [57, 58] could be used for its creation. As discussed in 4.2, a simple domain ontology based on human-readable files was defined and implemented for the developed prototype.

Scientific repositories. As described in Section 3, several domain-specific and general-purpose scientific repositories are available. Experimental data stored in these repositories can feed the operational database with new instances. Since the interpretation of experimental data is dynamically postponed, this phase only requires a "syntactic" and lightweight *wrapper* for each data source to match the operational database schema. This wrapper handles format and schema conversions to fit an experiment in the operational database's general schema, but does not need to semantically check, translate or verify experimental data. This process shall facilitate the management of different and new scientific repositories, separating the purely syntactic importing phase from the semantic-aware interpretation and usage phase.

Scholarly repositories. When experiments are extracted from scientific publications and each experiment keeps track of its original source, the integration of a scholarly data repository can aid many of the identified requirements. First of all, this integration avoids the manual input of detailed scholarly data when inserting a new experiment, since the experiment can just be associated with the publication identifier and related information is automatically retrieved. This avoids inconsistencies which could arise from inputting and storing this kind of information as more or less structured text, which currently happens in experimental repositories in this domain [5–8], such as naming ambiguities and missing data. The integration of scholarly data and associated *citation network analysis* techniques can aid the discovery of new and potentially relevant publications to acquire experiments based on publications which are the sources of experiments already in the operational database. Detailed and structured scholarly data support exploration and enable scholarly-related data aggregation, for example to aggregate analysis results for experiments published by the same author or journal. Currently, many open scholarly data repositories and analysis tools are available [35, 38], as initiatives to publish open bibliographic citation information as linked data, such as Open Citations [37].

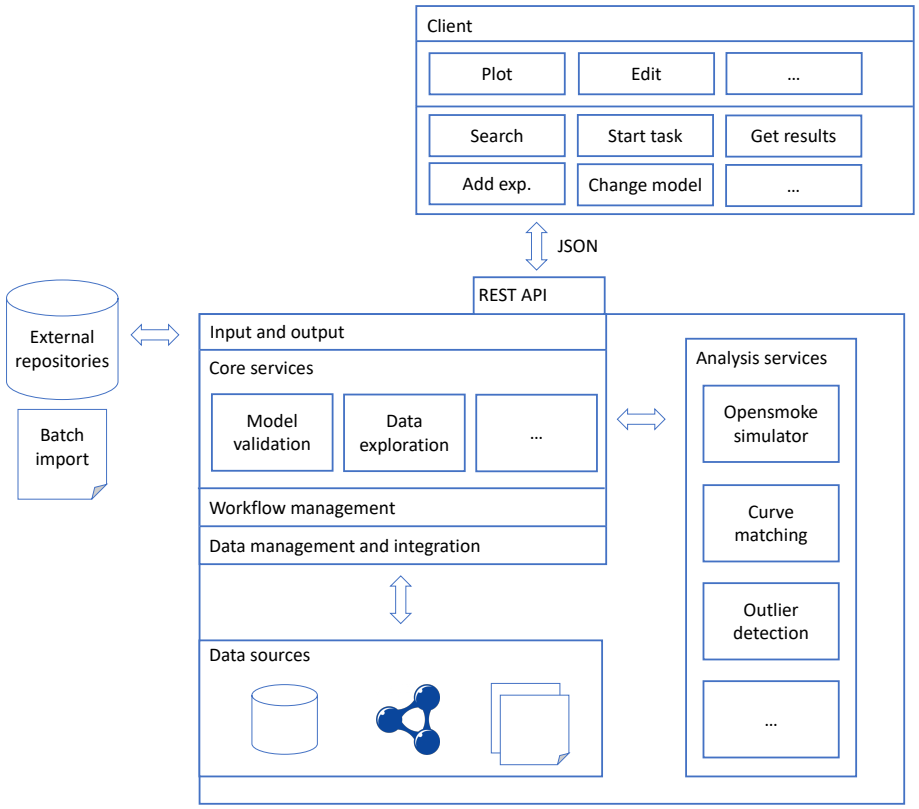

Fig. 10. Sketch of the architecture for the framework.

6.2. *Framework*

The main architectural components are illustrated in Figure 10.

A central server includes all data sources, core services, the input and output layer, external analysis services, workflow management and data management. The central server's design is largely based on a service-oriented architecture (SOA). Services are meant to be as independent as possible, keeping the communication layer simple and mostly mediated by the operational database. The latter, due to its general model and lazy interpretation of data, makes it possible to retrieve and store generic attributes associated with experiments and models or other attributes. This enables, for example, assigning a generic analysis result — with a service-defined name — as an attribute to a set of experiments or to an experiment/model pair. That result can later be retrieved by the same or other services. In this sense, with the exception of the shared operational database, the proposed architecture follows a *microservice* architecture design [59, 60], characterized by largely autonomous services and a lightweight communication layer. This has been proved to enhance interoperability, scalability, integration, and independent development with respect to traditional service-oriented architectures. These features are particularly important for the system's analysis services, which are those meant to be continuously developed, also by third parties, to enrich framework capabilities.

Core services provide the framework's core functions: model validation, data and results exploration, and so on. These services are designed to interface with analysis services on one side, which provide actual simulation, computational and analysis results, and with the data management and integration layer on the other side, which provides access to the data sources.

The *data management and integration* layer handles the integration of the different data sources described in Subsection 6.1 (including the operational database), data cleaning and quality management. It allows core services to perform complex, exploratory and quality-aware queries over them. This layer should be able to efficiently search, retrieve and integrate data based on the patterns of the acquisition/validation/exploration cycle. For example, retrieving all the experiments needed to validate a certain model change.

Analysis services provide a wide range of functions to compute validation indices, statistical analysis, aggregated values, and so on. Model simulation is managed as an analysis service. Each function is wrapped in a service standardizing its interface. This allows their uniform management and the storage of their outputs as attributes with an integrated schema in the operational database. Each analysis outcome is stored to be reused as needed, even as a sub-step in a more articulated task or by a different user. Analysis outcomes are stored in the operational database through a generalized attribute-based interface provided by a core service. Analysis services are designed to be developed also by third parties.

A *task* in the framework is defined over the core services which, in turn, can refer to the data management layer or to analysis services for specific sub-tasks. The *workflow management* architectural component handles functions such as resource allocation, task scheduling and parallelization. This component could be an existing workflow management system (see Section 3) integrated in the framework.

Given the variety of existing formats and standards, the *input and output* layer should integrate conversion tools. As discussed in Subsection 6.1, external scientific repositories need a syntactic wrapper for format conversion, even if semantic checks and conversions are not required at input time. The input interface enables inserting or modifying experiments and models, while the output interface allows accessing both experimental/model data and analysis outcomes. On top of the input/output layer, a REST API makes it possible to interact with the framework and export data as JSON.

The web client accesses the server through REST API and provides a user friendly way of interacting with the framework. This includes searching, starting a task, browsing the results, adding experiments or making changes in models. Moreover, it includes additional client-side functions, such as plotting, graph editing, and data editing.

7. Concluding remarks

The effective integration of large-scale scientific data analysis in the development cycle of scientific models, especially in their validation, is one of today's challenges, given the increasing availability of scientific open data and tools to automatically assess a model's performance. This is potentially allowing rapid and extremely significant technological advancements. Starting from the well-defined domain of kinetic modelling of combustion processes, this work takes a more general perspective, analyzing use cases and emerging requirements for an integrated framework which aims to automatically exploit large-scale scientific data analysis in the scientific development cycle, especially in the validation phase. An extended prototype was developed to better describe and refine emerging requirements and a preliminary architecture was designed, thus setting the basis for its refinement and development in future activities. This work mainly focuses on data-related requirements, since data integration, interpretation and validation are a pre-requisite for every subsequent analysis task.

Identified requirements lay the foundations for future research directions. Moreover, future work includes integration with existing third-party projects, in particular, workflow management systems and general purpose data repositories, the proposed framework's validation in other domains and more emphasis on analysis tasks which can be enabled by data management activities discussed in this paper. Analysis tasks should not only directly reflect users' requests, but also be automatically derived by the framework based on higher level user-defined goals. In this direction, a domain knowledge which facilitates a better interpretation of acquired data and enables automatic *reasoning* on it has a key role. The envisioned role for the human domain experts in this context sees them defining higher level tasks (for example, to "execute a new model on all existing experimental data necessary to validate a certain range of a variable") and, directly or indirectly, the knowledge necessary to interpret them. Future work also includes the proposed framework's integration with other techniques in the model development cycle. In particular, an effective integration with automatic model development techniques, which constitute a recent and very promising direction in the considered domain[61, 62], could highlight critical conditions that need further refinement in these models in a totally automatic development and validation cycle.

References

[1] L. H. Marcial and B. M. Hemminger, "Scientific data repositories on the web: An initial survey," *Journal of the American Society for Information Science and Technology*, vol. 61, no. 10, pp. 2029–2048, 2010.

[2] 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, *et al.*, "The FAIR guiding principles for scientific data management and stewardship," *Scientific data*, vol. 3, pp. 160018:1–9, 2016.

[3] A. de Waard, "Research data management at Elsevier: Supporting networks of data and workflows," *Information Services & Use*, vol. 36, no. 1-2, pp. 49–55, 2016.

[4] I. V. Pasquetto, B. M. Randles, and C. L. Borgman, "On the reuse of scientific data," *Data Science Journal*, vol. 16, no. 8, pp. 1–9, 2017.

[5] T. Varga, T. Turányi, E. Czinki, T. Furtenbacher, and A. Császár, "Respecth: a joint reaction kinetics, spectroscopy, and thermochemistry information system," in *Proceedings of the 7th European Combustion Meeting*, vol. 30, pp. 1–5, 2015.

[6] G. L. Goteng, N. Nettyam, and S. M. Sarathy, "Cloudflame: Cyberinfrastructure for combustion research," in *Information Science and Cloud Computing Companion (ISCC-C), 2013 International Conference on*, pp. 294–299, IEEE, 2013.

[7] B. W. Weber and K. E. Niemeyer, "ChemKED: A human-and machine-readable data standard for chemical kinetics experiments," *International Journal of Chemical Kinetics*, vol. 50, pp. 135–148, March 2018.

[8] M. Frenklach, "Transforming data into knowledge - Process informatics for combustion chemistry," *Proceedings of the combustion Institute*, vol. 31, no. 1, pp. 125–140, 2007.

[9] M. Bernardi, M. Pelucchi, A. Stagni, L. Sangalli, A. Cuoci, A. Frassoldati, P. Secchi, and T. Faravelli, "Curve matching, a generalized framework for models/experiments comparison: An application to n-heptane combustion kinetic mechanisms," *Combustion and Flame*, vol. 168, pp. 186–203, 2016.

[10] A. Rigamonti, "Automatic Modeling System: a database based infrastructure to develop, validate and evaluate scientific models. An application to combustion kinetic models, Graduation Thesis, Politecnico di Milano," 2017.

[11] G. Scalia, M. Pelucchi, A. Stagni, T. Faravelli, and B. Pernici, "Storing combustion data experiments: New requirements emerging from a first prototype," in *Semantics, Analytics, Visualization, - 3rd International Workshop, SAVE-SD 2017, Perth, Australia, April 3, 2017, and 4th International Workshop, SAVE-SD 2018, Lyon, France, April 24, 2018, Revised Selected Papers, LNCS, volume 10959* (A. González-Beltrán, F. Osborne, S. Peroni, and S. Vahdati, eds.), (Cham), pp. 138–149, Springer International Publishing, 2018.

[12] A. Cuoci, A. Frassoldati, T. Faravelli, and E. Ranzi, "Opensmoke++: An object-oriented framework for the numerical modeling of reactive systems with detailed kinetic mechanisms," *Computer Physics Communications*, vol. 192, pp. 237–264, 2015.

[13] B. Wang, H. Olivier, and H. Grönig, "Ignition of shock-heated h2-air-steam mixtures," *Combustion and flame*, vol. 133, no. 1-2, pp. 93–106, 2003.

[14] H. Wang and D. A. Sheen, "Combustion kinetic model uncertainty quantification, propagation and minimization," *Progress in Energy and Combustion Science*, vol. 47, pp. 1–31, 2015.

[15] C. Olm, I. G. Zsély, R. Pálvölgyi, T. Varga, T. Nagy, H. J. Currand, and T. Turányi, "Comparison of the performance of several recent hydrogen combustion mechanisms," *Combustion and Flame*, vol. 161, no. 9, pp. 2219–2234, 2014.

[16] C. Olm, I. G. Zsély, T. Varga, H. J. Currand, and T. Turányi, "Comparison of the performance of several recent syngas combustion mechanisms," *Combustion and Flame*, vol. 162, no. 5, pp. 1793–1812, 2015.
[17] C. Olm, I. G. Zsély, R. Pálvölgyi, T. Varga, T. Nagy, H. J. Curran, and T. Turányi, "Comparison of the performance of several recent hydrogen combustion mechanisms," *Combustion and Flame*, vol. 161, no. 9, pp. 2219–2234, 2014.
[18] A. Stagni, A. Frassoldati, A. Cuoci, T. Faravelli, and E. Ranzi, "Skeletal mechanism reduction through species-targeted sensitivity analysis," *Combustion and Flame*, vol. 163, pp. 382–393, 2016.
[19] C. Cavallotti, M. Pelucchi, and S. Klippenstein, "EStokTP: Electronic structure to temperature and pressure dependent rate constants," *unpublished*, 2017.
[20] M. Keçeli, Y.-P. L. Elliott, M. Johnson, C. Cavallotti, Y. Georgievskii, W. P. Green, J. Wozniak, A. M. Jasper, and S. Klippenstein, "Automated computational thermochemistry for butane oxidation: A prelude to predictive automated combustion kinetic," *Proceedings of the Combustion Institute, in press*, 2018.
[21] J. Zeng and K. W. Glaister, "Value creation from big data: Looking inside the black box," *Strategic Organization*, vol. 16, no. 2, pp. 105–140, 2018.
[22] I. V. Pasquetto, A. E. Sands, P. T. Darch, and C. L. Borgman, "Open data in scientific settings: From policy to practice," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 1585–1596, ACM, 2016.
[23] C. L. Borgman, *Big data, little data, no data: Scholarship in the networked world.* MIT Press, 2015.
[24] H. Jagadish, "Big data and science: Myths and reality," *Big Data Research*, vol. 2, no. 2, pp. 49–52, 2015.
[25] N. Hansen, X. He, R. Griggs, and K. Moshammer, "Knowledge generation through data research: New validation targets for the refinement of kinetic mechanisms," *Proceedings of the Combustion Institute*, 2018.
[26] D. Ardagna, C. Cappiello, W. Samá, and M. Vitali, "Context-aware data quality assessment for big data," *Future Generation Computer Systems*, vol. 89, pp. 548–562, 2018.
[27] C. Cappiello, W. Samá, and M. Vitali, "Quality awareness for a successful big data exploitation," in *Proceedings of the 22nd International Database Engineering & Applications Symposium*, pp. 37–44, ACM, 2018.
[28] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, "Data cleaning: Overview and emerging challenges," in *Proceedings of the 2016 International Conference on Management of Data*, pp. 2201–2206, ACM, 2016.
[29] A. Chalamalla, I. F. Ilyas, M. Ouzzani, and P. Papotti, "Descriptive and prescriptive data cleaning," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 445–456, ACM, 2014.
[30] A. Abelló, O. Romero, T. B. Pedersen, R. Berlanga, V. Nebot, M. J. Aramburu, and A. Simitsis, "Using semantic web technologies for exploratory OLAP: a survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 2, pp. 571–588, 2015.
[31] C. S. Liew, M. P. Atkinson, M. Galea, T. F. Ang, P. Martin, and J. I. V. Hemert, "Scientific workflows: Moving across paradigms," *ACM Computing Surveys (CSUR)*, vol. 49, no. 4, p. 66, 2017.
[32] M. Atkinson, S. Gesing, J. Montagnat, and I. Taylor, "Scientific workflows: Past, present and future," *Future Generation Computer Systems*, vol. 75, pp. 216 – 227, 2017.
[33] M. A. Rodriguez and R. Buyya, "A taxonomy and survey on scheduling algorithms for scientific workflows in iaas cloud computing environments," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 8, p. e4041, 2017.
[34] 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," *The International Journal of High Performance Computing Applications*, vol. 32, no. 1, pp. 159–175, 2018.
[35] F. Xia, W. Wang, T. M. Bekele, and H. Liu, "Big scholarly data: A survey," *IEEE Transactions on Big Data*, vol. 3, no. 1, pp. 18–35, 2017.
[36] D. MacMillan, "Data sharing and discovery: What librarians need to know," *The Journal of Academic Librarianship*, vol. 40, no. 5, pp. 541–549, 2014.
[37] S. Peroni, D. Shotton, and F. Vitali, "One year of the opencitations corpus," in *International Semantic Web Conference*, pp. 184–192, Springer, 2017.
[38] C. Dunne, B. Shneiderman, R. Gove, J. Klavans, and B. Dorr, "Rapid understanding of scientific paper collections: Integrating statistics, text analytics, and visualization," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 12, pp. 2351–2369, 2012.
[39] F. Osborne, E. Motta, and P. Mulholland, "Exploring scholarly data with rexplore," in *International semantic web conference*, pp. 460–477, Springer, 2013.
[40] P. Ristoski and H. Paulheim, "Semantic web in data mining and knowledge discovery: A comprehensive survey," *Web semantics: science, services and agents on the World Wide Web*, vol. 36, pp. 1–22, 2016.
[41] L. Libkin, W. Martens, and D. Vrgoč, "Querying graphs with data," *Journal of the ACM (JACM)*, vol. 63, no. 2, pp. 14–53, 2016.
[42] 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," *Semantic Web*, vol. 8, no. 3, pp. 471–487, 2017.
[43] A. Schätzle, M. Przyjaciel-Zablocki, S. Skilevic, and G. Lausen, "S2RDF: RDF querying with SPARQL on spark," *Proceedings of the VLDB Endowment*, vol. 9, no. 10, pp. 804–815, 2016.

[44] W. Pejpichestakul, E. Ranzi, M. Pelucchi, A. Frassoldati, A. Cuoci, A. Parente, and T. Faravelli, "Examination of a soot model in premixed laminar flames at fuel-rich conditions," *Proceedings of the Combustion Institute, in press*, 2018.
[45] B. Berger, N. M. Daniels, and Y. W. Yu, "Computational biology in the 21st century: Scaling with compressive algorithms," *Communications of the ACM*, vol. 59, no. 8, p. 72, 2016.
[46] J. Rung and A. Brazma, "Reuse of public genome-wide gene expression data," *Nature Reviews Genetics*, vol. 14, no. 2, p. 89, 2013.
[47] R. Margolis, L. Derr, M. Dunn, M. Huerta, J. Larkin, J. Sheehan, M. Guyer, and E. D. Green, "The national institutes of health's big data to knowledge (bd2k) initiative: capitalizing on biomedical big data," *Journal of the American Medical Informatics Association*, vol. 21, no. 6, pp. 957–958, 2014.
[48] K. Dolinski and O. G. Troyanskaya, "Implications of big data for cell biology," *Molecular biology of the cell*, vol. 26, no. 14, pp. 2575–2578, 2015.
[49] L. Aagesen, J. Adams, J. Allison, W. Andrews, V. Araullo-Peters, T. Berman, Z. Chen, S. Daly, S. Das, S. DeWitt, *et al.*, "Prisms: An integrated, open-source framework for accelerating predictive structural materials science," *JOM*, vol. 70, no. 10, pp. 2298–2314, 2018.
[50] H. Gossler, L. Maier, S. Angeli, S. Tischer, and O. Deutschmann, "Carmen: a tool for analysing and deriving kinetics in the real world," *Physical Chemistry Chemical Physics*, vol. 20, no. 16, pp. 10857–10876, 2018.
[51] N. Di Blas, M. Mazuran, P. Paolini, E. Quintarelli, and L. Tanca, "Exploratory computing: a comprehensive approach to data sensemaking," *International Journal of Data Science and Analytics*, vol. 3, no. 1, pp. 61–77, 2017.
[52] R. Yu, U. Gadiraju, B. Fetahu, and S. Dietze, "Adaptive focused crawling of linked data," in *International Conference on Web Information Systems Engineering*, pp. 554–569, Springer, 2015.
[53] A. Cohan and N. Goharian, "Scientific article summarization using citation-context and article's discourse structure," *arXiv preprint arXiv:1704.06619*, 2017.
[54] A. Wasay, M. Athanassoulis, and S. Idreos, "Queriosity: Automated data exploration," in *2015 IEEE International Congress on Big Data, New York City, NY, USA, June 27 - July 2, 2015* (B. Carminati and L. Khan, eds.), pp. 716–719, IEEE, 2015.
[55] M. A. Soliman, I. F. Ilyas, and K. C.-C. Chang, "Top-k query processing in uncertain databases," in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pp. 896–905, IEEE, 2007.
[56] Z. Xiang, J. Zheng, Y. Lin, and Y. He, "Ontorat: automatic generation of new ontology terms, annotations, and axioms based on ontology design patterns," *Journal of biomedical semantics*, vol. 6, no. 1, p. 4, 2015.
[57] S. Spangler, A. D. Wilkins, B. J. Bachman, M. Nagarajan, T. Dayaram, P. Haas, S. Regenbogen, C. R. Pickering, A. Comer, J. N. Myers, *et al.*, "Automated hypothesis generation based on mining scientific literature," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1877–1886, ACM, 2014.
[58] K. Gábor, H. Zargayouna, I. Tellier, D. Buscaldi, and T. Charnois, "A typology of semantic relations dedicated to scientific literature analysis," in *International Workshop on Semantic, Analytics, Visualization*, pp. 26–32, Springer, 2016.
[59] N. Alshuqayran, N. Ali, and R. Evans, "A systematic mapping study in microservice architecture," in *2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA)*, pp. 44–51, IEEE, 2016.
[60] N. Kratzke and P.-C. Quint, "Understanding cloud-native applications after 10 years of cloud computing-a systematic mapping study," *Journal of Systems and Software*, vol. 126, pp. 1–16, 2017.
[61] C. W. Coley, R. Barzilay, T. S. Jaakkola, W. H. Green, and K. F. Jensen, "Prediction of organic reaction outcomes using machine learning," *ACS central science*, vol. 3, no. 5, pp. 434–443, 2017.
[62] C. W. Coley, W. H. Green, and K. F. Jensen, "Machine learning in computer-aided synthesis planning," *Accounts of chemical research*, vol. 51, no. 5, pp. 1281–1289, 2018.