mUQSA - an online service for Uncertainty Quantification and Sensitivity Analysis

Bartosz Bosak[1 [0000−0002−9331−3166]], Piotr Kopta[1 [0000−0001−8237−0969]], Michał Kulczewski[1 [0000−0002−1349−2927]], and Tomasz Piontek[1 [0000−0003−0147−3996]]

Poznan Supercomputing and Networking Center
ul. Jana Pawła II 10, 61-139 Poznan, Poland
office@man.poznan.pl
https://www.psnc.pl

Abstract. Enabling uncertainty quantification (UQ) for a computational model can be a complex process, often exceeding the domain expertise of its developer or user. To address this challenge, we developed the mUQSA platform, which streamlines typical uncertainty quantification scenarios through a modern web-based environment and pre-configured automated workflows that leverage large-scale computing resources for state-of-the-art UQ techniques. mUQSA is designed with two key objectives: to support users in analyzing how input uncertainties propagate through their models and to identify the input parameters that most significantly influence outcomes through sensitivity analysis (SA). The particular strength of mUQSA lies in its accessibility: offered as a Software-as-a-Service (SaaS) solution, mUQSA eliminates the need for installation or configuration of software, enabling users to access UQ/SA methods directly from a web browser. Its user-friendly, intuitive interfaces for defining use cases and displaying results guarantee high accessibility, while a robust execution layer, supported by HPC resources, ensures mUQSA can handle even very complex computational models.

Keywords: Uncertainty Quantification · Sensitivity Analysis · multiscale · HPC

1 Introduction

Uncertainty Quantification is one of the major tools for assessing credibility of computational simulations. Based on a solid mathematical framework, UQ techniques address several problems in scientific modeling, such as certification, prediction, verification and validation of model and software, parameter estimation, data assimilation, and inverse problems [20]. However, applying uncertainty quantification (UQ) to computational models, although often essential for ensuring actionable results, represents an additional layer of effort, both logically and technologically, that extends beyond the core domain of expertise of an individual scientist or engineer. Furthermore, UQ algorithms, while potentially critical to the validity of simulation results, are supplementary to the fundamental scientific model itself. Finally, the increased number of executions needed for reliable

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

UQ imposes significantly higher computational resource demands, especially for highly complex models, such as those in multiscale simulations.

Given these factors, we leveraged the EasyVVUQ library [18], which provides core methods for efficient Verification, Validation, and Uncertainty Quantification of computational models and is a key outcome of the VECMA project [23], alongside two components of the QCG family [17]: QCG-Portal, engineered for the intuitive submission and management of computational experiments on supercomputers, and QCG-PilotJob [16], a second-level scheduler that optimizes the execution of numerous logical jobs within a single HPC resource allocation, to build the multipurpose Uncertainty Quantification and Sensitivity Analysis framework, mUQSA. To provide seamless access to the advanced computational capabilities it integrates, mUQSA has been developed to work in a Software-as-a-Service (SaaS) model.

Established under the Polish national PIONIER-LAB project [15], mUQSA is currently being refined within the European HiDALGO2 project [11] to address the specific requirements of environmental use cases and support the evolving needs of simulation developers and end users.

From a design perspective, mUQSA integrates intuitive web-based user interface, HPC processing, and state-of-the-art UQ algorithms, delivering a comprehensive and versatile environment for widely requested uncertainty related studies, including non-intrusive forward uncertainty quantification and sensitivity analysis.

That said, mUQSA is not intended to be a universal tool for Uncertainty Quantification (UQ). Instead, it targets common UQ and SA workflows, aiming to streamline their implementation. The provided UQ methods enable users to identify basic trends and compute key statistical moments for Quantities of Interest (QoIs), which are critical output values influenced by uncertain input parameters. Furthermore, the Sensitivity Analysis, based on Sobol's indices, offers a robust approach to assessing the relative importance of specific inputs. As such, the platform offers flexibility across a broad spectrum of applications, ranging from monolithic molecular dynamics codes and complex multiscale climate models to the development of digital human twins.

Thanks to the intuitiveness of the interface and the automation offered, mUQSA can also be a starting point to get practical knowledge in the fields of uncertainty quantification and sensitivity analysis for those who are not yet familiar with the concepts.

The rest of this paper is organised as follows. In Section 2 we briefly present other software available for similar purposes. Section 3 outlines main functionality provided by mUQSA and the way it is offered to users. Section 4 elaborates on the execution layer of the platform. Finally, in Section 5 we conclude and present future plans for the development of the platform.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

2 Related work

Verification, Validation, and Uncertainty Quantification (VVUQ) are critical topics addressed in numerous projects and initiatives, playing an essential role in both science and industry. To support the application of VVUQ in scientific simulations, efforts are underway, such as those of the American Society of Mechanical Engineers (ASME) [22], to establish best practices and develop standards for common VVUQ processes. SmartUQ, a leading provider of UQ software, has also highlighted the benefits of simplification, reporting that its UQ tools have saved stakeholders millions of dollars and thousands of working hours [1]. Alongside such commercial solutions, a number of smaller-scale initiatives and diverse research teams successfully develop software tailored to general and specialized VVUQ applications, much of it distributed as open source. Libraries such as EasyVVUQ [18], UncertainPy [21], Chaospy [8], and OpenTURNS [3], as well as more comprehensive toolkits and frameworks such as Dakota [6], Uranie [4], and the VECMA Toolkit [9], offer a high degree of flexibility and adaptability in a wide range of use cases. However, these tools often require substantial initial effort to set up and may encounter scalability challenges in large-scale workflows without complex configuration. Specifically, while Dakota is one of the most popular and feature-rich UQ frameworks, offering advanced capabilities like calibration and surrogate modeling, its setup process can be cumbersome. Configuring Dakota to run workflows on remote computing resources involves minimal automation, which presents significant challenges to novice users. Similar difficulties are encountered by users of other well-recognized tools, such as OpenTURNS. The situation is somewhat different with the VECMA Toolkit, where FabSim3[10], combined with EasyVVUQ and QCG-PilotJob[16], introduces several automation features to facilitate the execution of UQ processes on HPC resources. However, VECMA Toolkit is focused on command-line usage, which may still be considered too complex for some users.

3 mUQSA Functionality

The goal of the mUQSA platform is to provide an easy-to-use web environment that enables automated uncertainty quantification and sensitivity analysis of computational models on high-performance computing (HPC) resources.

This placement of the mUQSA toolkit is achieved through several means.

Intuitive wizard for the preparation of the UQ/SA scenario. The wizard is divided into several logical steps that allow one to collect all necessary information required to run UQ/SA for a selected model on computational resources. Thus, we have separate steps for the definition of sampled parameters, configuration of the method (see Figure 1), or specification of the way the model should be executed. Combined with the provided documentation, the step-by-step collection of information in the wizard streamlines the process of incorporating UQ into the model and makes it achievable even for non-experienced users.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

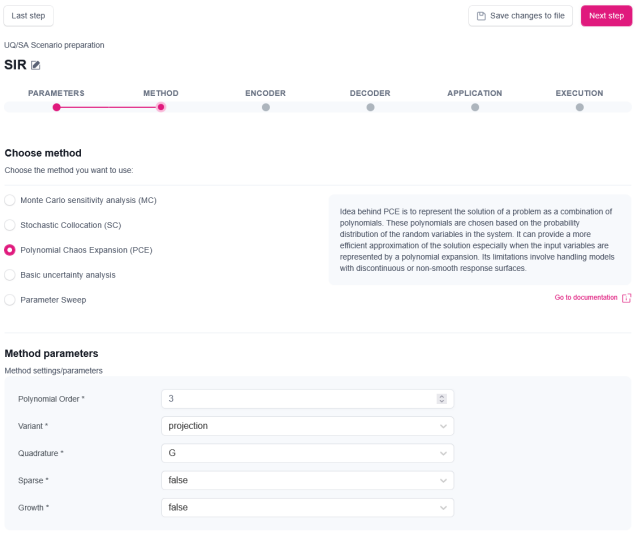

Fig. 1. Selection and configuration of a UQ method in the mUQSA wizard

Built-in interactive presentation of results With built-in support for displaying output data as interactive charts and tables in predefined formats, users can access insights from their analysis in a concise, visually engaging way, with the structure effectively showcasing key findings. Figure 2 illustrates a result presentation view for the sensitivity analysis, using the SIR model as an example (this model will be illustrated in more detail later in Section 5). If this is not sufficient, users can also access the raw results data, enabling deeper analysis when needed.

Automated execution on HPC machine One of the primary challenges in uncertainty quantification for computationally intensive models is the ability to efficiently and transparently perform the necessary, often extensive, computations on supercomputers. This is where mUQSA excels. The platform seamlessly manages the scheduling and execution of tasks required for the sampled parameters, reducing the user's role to simply initiating the experiment, while all subsequent processes run effortlessly in the background.

Support and documentation While mUQSA simplifies the definition and execution of uncertainty quantification and sensitivity analysis tasks, users with limited prior knowledge of these procedures may find the interaction with the platform challenging. Additionally, some advanced features may require more in-depth explanation beyond the brief help descriptions provided in the graphical user interfaces (GUIs). To address this, mUQSA is supported by comprehensive documentation and a set of tutorials that demonstrate its usage in detail, ensuring that users receive the necessary assistance. [14]

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

Fig. 2. Presentation of Sensitivity Analysis results in mUQSA

3.1 mUQSA workflow

Conceptually, mUQSA can be divided into two main layers: the User Interface (UI) layer, and the Execution layer.

The UI layer has two main duties:

1. definition of all input parameters and configuration settings for UQ/SA scenario,
2. presentation of results for further analysis, once the required calculations have been completed.

In turn, the Execution layer's role is to efficiently execute required computations according to the selected method and user-defined input data. Given this context, an overall picture of the execution flow in mUQSA can be visualised as in Figure 3

The Scenario Definition and Analysis steps correspond directly to the two above duties of UI. Scenario Definition's aim is to collect all the specific data on the scenario from a user, including the selection of a Method and the definition of Parameters and then to push it for fully automatic processing on an HPC machine. Analogically, the Analysis step receives results from the processing on the HPC machine and displays an interactive web page with condensed information for further analysis by a user.

The rest of steps of the workflow have a strictly computational character and belong to the Execution layer. Sampler generates sets of inputs for model evaluations according to a selected method. Then, multiple streams, each comprising an Encoder, Execution, and Decoder, are instantiated - one per required evaluation - and executed in parallel. Next, the output data from the evaluations is merged together by Collator and analysed programmatically by Analyser, again

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

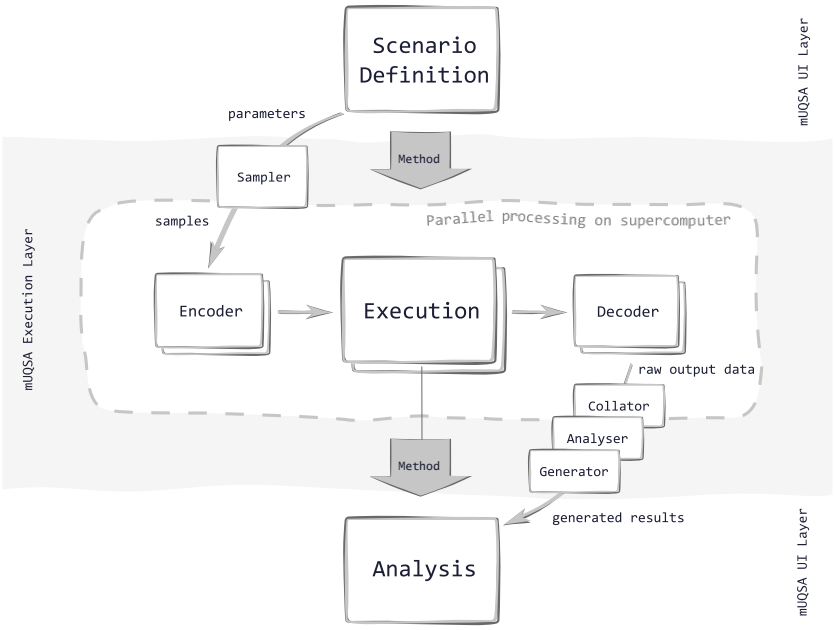

Fig. 3. High-level view on the mUQSA conceptual workflow

in accordance with the selected method. Finally, the results are converted into human-readable form by Generator.

3.2 Supported UQ/SA methods

mUQSA's core algorithmic component is derived from the EasyVVUQ library [18]. This choice of algorithmic engine for mUQSA is deliberate, as EasyVVUQ, with its well-defined API, provides a robust set of widely recognized and valued uncertainty quantification methods. mUQSA incorporates most of these techniques, including:

- **(Quasi-)Monte Carlo (MC)** - a statistical method for numerical integration or simulation that uses random sampling. Quasi-Monte Carlo improves on standard Monte Carlo by using low-discrepancy sequences to achieve faster convergence.
- **Polynomial Chaos Expansion (PCE)** - a technique that represents uncertain quantities as a series of orthogonal polynomials, providing an efficient way to approximate the response of a system to input uncertainties.
- **Stochastic Collocation (SC)** - a spectral method that approximates the solution of a problem by collocating the system at specific sample points (based on the probability distribution of the inputs) and constructing a global approximation through interpolation

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

- **Ensemble Analysis** - a basic method where multiple simulations are run with different sets of input parameters to analyze the overall system behavior. The results are aggregated to understand the variability and uncertainty.
- **Parameter Sweep** - a technique where a model is run repeatedly across a range of input values, systematically varying one or more parameters to explore how the system's output changes with respect to these inputs.

As a result, mUQSA can be viewed, with some simplification, as a web-based interface for EasyVVUQ. Specifically, it serves two key purposes: 1) provides an intuitive interface to access selected EasyVVUQ methods, and 2) is preconfigured to use a designated computational resource, ensuring efficient execution of the computations defined by these methods.

Each of the methods supported by mUQSA can handle scalar and vector Quantities of Interest (QoIs) and can be configured to match specific needs of the given scenario.

Additionally, leveraging the results of the MC, PCE, and SC methods, the platform automatically computes **Sobol's indices**. This allows for a detailed sensitivity analysis that quantifies the contribution of each input parameter (or combinations of parameters) to the output variance. The Sobol indices help identify the most influential factors driving uncertainty by decomposing the total variance into contributions from individual parameters and their interactions.

It is worth noting that the presentation of results in the platform is largely consistent across methods, enabling users to compare them, discern trends, and identify the most suitable options.

4 mUQSA Architecture

Developing a comprehensive Uncertainty Quantification system poses challenges on multiple conceptual and technological fronts. Fortunately, in development of mUQSA, we were able to reuse several already available components that allowed to build a functional system in relatively short time.

The conceptual scheme of the mUQSA workflow described in Figure 3 can be mapped to the representation focused on the concrete software components shown in Figure 4. QCG, EasyVVUQ, and QCG-PilotJob [16] are software packages adopted from previous initiatives, while all the blocks highlighted in red are newly developed components for mUQSA. The following sections will detail this flow, focusing on the specific software components involved.

Problem definition: mUQSA GUI

The mUQSA interface is built on the capabilities of the QCG software, particularly QCG-Portal, which provides core mechanisms for executing and managing computational tasks on remote resources directly from a web browser. QCG-Portal also supports interface customization for specific use cases, enabling adjustments to task input data specification and execution monitoring to meet

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

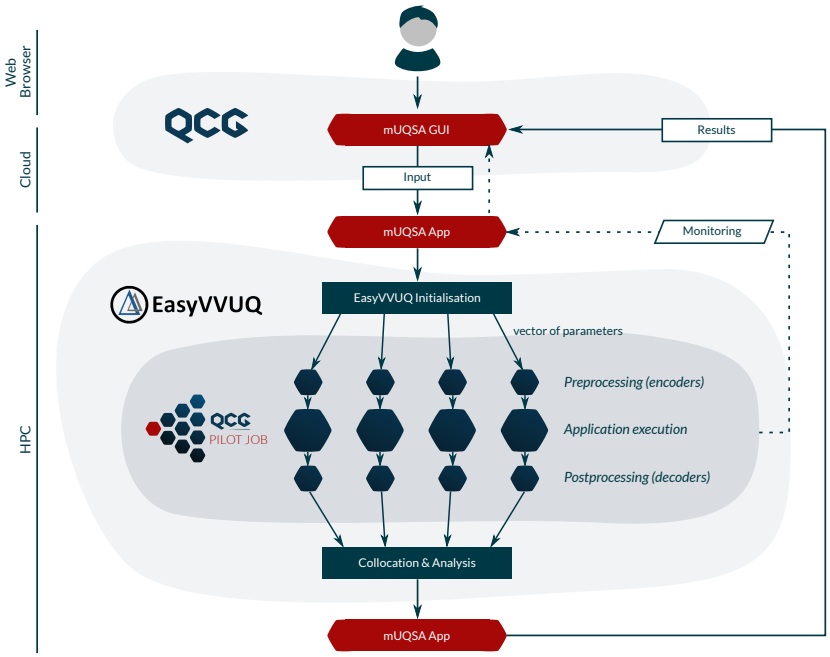

Fig. 4. High-level architecture of mUQSA platform

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

particular needs. Consequently, the mUQSA wizard is a custom JavaScript application that integrates and leverages core functionality of the portal. Once all input data is collected through the wizard, mUQSA transfers it to QCG-Portal, which submits a specific job to be executed on an HPC resource. Importantly, users can track the execution state through a dedicated mUQSA monitoring panel within the task details view in QCG-Portal (see Figure 5).

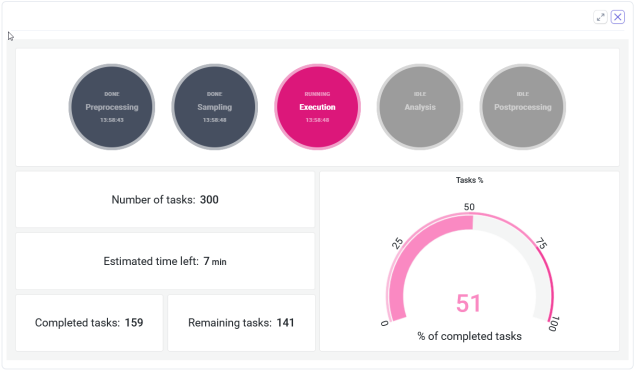

Fig. 5. mUQSA monitoring dashboard in QCG-Portal

Execution preparation: mUQSA App

When the job submitted from QCG-Portal is launched on an HPC resource, the specifically crafted mUQSA App is started and takes care of establishing the entire execution. In the first step, it converts input configuration (stored in a JSON document) to the EasyVVUQ's Campaign object to enable seamless execution of the scenario with the selected method on allocated resources.

Execution: EasyVVUQ and QCG-PilotJob

The configured campaign is executed by EasyVVUQ through a series of steps, several of which can be effectively parallelized. After method-specific sampling, the resulting parameter vectors are used to evaluate the model, which also involves encoding and decoding phases. Using EasyVVUQ's integration with QCG-PilotJob, these evaluations are dynamically scheduled and executed in parallel, optimizing resource utilization and improving computational efficiency. In particular, since QCG-PilotJob supports the execution of parallel codes, individual evaluations can be applications parallelized with MPI or OpenMP. Moreover, beyond evaluating pre-installed codes, mUQSA enables the execution of models provided as Apptainer containers [2], relieving users from the often challenging task of compiling and installing code on a computing resource. Throughout the

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

process, custom monitoring scripts relay updates on the execution stage and progress to centralized QCG monitoring services. Upon completion, EasyVVUQ aggregates data from all evaluations and performs a method-specific analysis.

Results preparation: mUQSA App

The raw output data generated by EasyVVUQ is then accessed by the mUQSA App, which transforms it into interactive and visually appealing web pages summarizing the analysis performed. Although the primary use case for mUQSA involves the GUI, the underlying software can also be used without it. Since the input data is in JSON format and web page generation occurs on the cluster, command-line usage is also possible.

Results visualisation: mUQSA GUI

Finally, the generated web page is presented to the user within QCG-Portal, enabling analysis of the results through zoomable and draggable charts, filtered tables, and, if needed, the option to export data into widely recognized formats for further exploration in an external program of choice.

5 Practical Example

This section demonstrates mUQSA's functionality through a case study of the SIR compartment model [24], a framework for simulating infections by categorizing a population into Susceptible (S), Infectious (I), and Recovered (R) groups [24]. The purpose of this section is to provide a brief overview, while more detailed showcases are available as tutorials on the mUQSA web page [14].

The SIR model starts with a defined state of S, I and R groups, and then it is iteratively updated using differential equations based on two key parameters:

β : Infection rate, the average number of contacts per person per time unit resulting in transmission from Susceptible to Infected.
γ : Recovery rate, the rate at which Infected individuals transition to Recovered (the inverse of infection duration).

The objective is to evaluate the output uncertainty of the model given uncertain inputs β and γ and analyse the model's sensitivity to their variability. The process is described below.

5.1 Accessing mUQSA

To use mUQSA, users require access to a resource where the framework is preconfigured. This may be a private, self-hosted deployment of QCG-Portal and mUQSA software or, more commonly, a public HPC system, such as the Proxima cluster at PSNC, with mUQSA and its dependencies pre-installed. Once access is granted, users can interact with mUQSA directly via a web browser,

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

eliminating the need for local software installation or configuration. By logging into QCG-Portal (e.g. https://qcg.pcss.plcloud.pl) with the credentials provided, users select mUQSA from the available applications - often the default in single-application setups - and launch the mUQSA scenario definition wizard.

5.2 Configuring UQ/SA Scenario

In the mUQSA wizard, users can choose to start a completely new scenario or load a previously prepared one and adjust its settings. The wizard consists of the following steps:

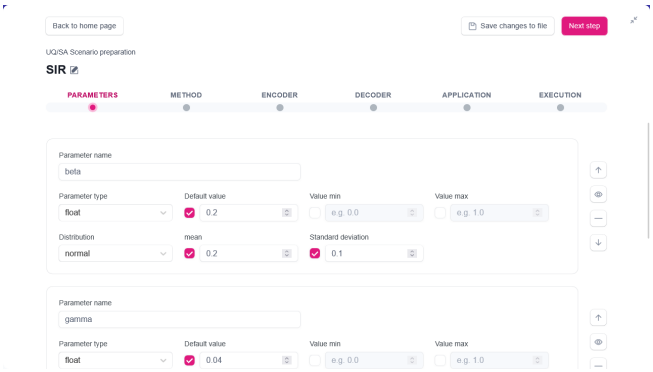

Fig. 6. Definition of parameters in the mUQSA wizard

Step 1 - Parameters : Allows users to define input parameters for analysis, specifically uncertain parameters, their distributions, and constraints. This guides UQ algorithms on how to sample parameter combinations for model evaluation. For our example analysis of the SIR model, β (infection rate) is assigned a uniform distribution, and γ (recovery rate) follows a normal distribution (Figure 6). The rest of parameters, such as the initial number of infected persons, have static values and would not be sampled.

Step 2 - Method : Enables the selection and configuration of the uncertainty quantification (UQ) method for the specified scenario. The wizard provides brief descriptions of the methods available to guide selection, with comprehensive documentation linked for new users seeking a deeper understanding. For our analysis, we chose the PCE method with a polynomial order of 3 (see Figure 1), although other methods can also be explored.

Step 3 - Encoder : Focuses on specifying how the generic input data from mUQSA, processed via EasyVVUQ, is encoded into a format compatible with the model. Our code relies on simple text file inputs, which can be generated using available encoders such as GenericEncoder. This encoder inserts sampled parameters into designated placeholders within a provided template

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

to produce the final input file. Simple templates can be specified directly in the portal, while more complex templates can be referenced from the file system of the computing resource. For highly specialized cases, custom encoders can be developed to meet specific requirements.

Step 4 - Decoder : Converts model-specific outputs (quantities of interest, QoI) into a generic format for mUQSA (EasyVVUQ) analysis. For the SIR model's CSV output, we use one of already available decoders, which is CSVDecoder. In our case, we analyze the Infected (I) QoI, though Susceptible (S) or Recovered (R) could also be examined. Custom decoders can be provided for complex cases.

Step 5 - Application : Selects the model for evaluation, which can be a preinstalled application or an Apptainer image. In both cases, mUQSA assumes that the model is available on a computing resource before the UQ workflow is launched. The SIR model, since it is bundled with mUQSA, is chosen by pointing to its location in the mUQSA installation directory on the Proxima cluster.

Step 6 - Execution : Specifies execution settings, including whether evaluations run serially or in parallel and how many evaluations run concurrently. For the SIR model, we configure serial execution with multiple evaluations running in parallel. To enhance user experience, the wizard estimates the required wall clock time and CPU hours based on the chosen UQ algorithm and specified execution settings.

Once configured, the scenario is ready to be submitted for processing on the remote computing resource.

5.3 Execution

The submitted mUQSA workflow is placed in a queue in the computing cluster, waits for free resources, and is then launched and executed. This process is fully automatic. Users can track the state and progress of the workflow using the built-in QCG-Portal features and the mUQSA monitoring panel provided (see Figure 5 for reference).

5.4 Analysis

When the calculations are complete, the results are displayed on a customised webpage accessible via QCG-Portal (see Figure 2). The output includes graphs and tables showing key statistics (e.g., mean, percentiles) and Sobol indices, tailored to scalar or vector QoIs. For the SIR model vector QoI (Infected, I), a line graph displays the number of infected individuals over time; scalar QoIs would use a pie chart. If the default presentation is insufficient, users can download raw data for further analysis in external tools.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

6 Conclusion and Future Work

Developing versatile, scalable and user-friendly software for uncertainty quantification can be extremely challenging if not infeasible. For this reason, in the initial phase of the development of mUQSA, we chose to temper expectations with respect to versatility. Our focus was to implement a curated set of widely recognized methods and deliver them through an intuitive web portal, ensuring efficient and transparent execution of required computations on supercomputers. The software is currently deployed on the PSNC's infrastructure and is readily accessible to interested users. To date, it has supported several application use cases (for example, [13]) and was made available to participants of a hackathon jointly organized by HiDALGO2, CIRCE, and SEAVEA projects [12]. At this point we already can confirm validity of our approach and its usefulness for the individuals and communities searching for easily-applicable UQ.

Our future plans include widely understood simplification and hardening of the platform, so it will be even more accessible, predictable, and adaptable to individual use-cases. As a first step, we aim to enhance its resilience against failures and introduce the ability to save and load EasyVVUQ Campaign objects at various stages of the workflow execution. These changes are essential for large use cases, where repetition of evaluations may be extremely inefficient.

Although our current focus is not on expanding the number or variety of supported UQ methods, we plan to extend this aspect in the future. Among other enhancements, we are considering adding Markov Chain Monte Carlo (MCMC) to address inverse problems, alongside exploring methods to enable the automatic construction of surrogate models. The implementation of these or other new techniques will depend largly on the expressed needs of the scientists and engineers.

Ultimately, the scope and direction of mUQSA's evolution will hinge on the extent to which experts from various branches of UQ science contribute to its development. To foster this, our planned efforts include promoting and disseminating mUQSA to facilitate outreach and forge new collaborations. We anticipate that the expertise of HiDALGO2, SEAVEA [19], and possibly other UQ-focused initiatives, such as DATAHYKING [5] and ELLIS [7], will advance the functionality and usability of the mUQSA framework, enabling it to address the requirements of transformative applications, such as AI-driven models.

Acknowledgments. Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (EuroHPC JU) and Poland, Germany, Spain, Hungary, France and Greece under grant agreement number: 101093457 (HiDALGO2 project). The development of mUQSA platform received partial funding from the PIONIER-LAB project (POIR.04.02.00-30-A005/16-00).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

References

1. 3545 University Ave Madison: SmartUQ About. https://www.smartuq.com/about/ (2025), [Online; accessed 22-April-2025]
2. Apptainer: Apptainer webpage. https://apptainer.org/ (2025), [Online; accessed 22-April-2025]
3. Baudin, M., Dutfoy, A., Iooss, B., Popelin, A.L.: OpenTURNS: An Industrial Software for Uncertainty Quantification in Simulation, p. 1–38. Springer International Publishing (2015). https://doi.org/10.1007/978-3-319-11259-6_64-1
4. Blanchard, Jean-Baptiste, Damblin, Guillaume, Martinez, Jean-Marc, Arnaud, Gilles, Gaudier, Fabrice: The uranie platform: an open-source software for optimisation, meta-modelling and uncertainty analysis. EPJ Nuclear Sci. Technol. **5**, 4 (2019). https://doi.org/10.1051/epjn/2018050, https://doi.org/10.1051/epjn/2018050
5. DATAHYKING: Marie curie doctoral network funded by the european commission under grant agreement no. 101072546, https://datahyking.eu/, [Online; accessed 22-April-2025]
6. Eldred, M., Bohnhoff, W., Hart, W.: Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, sensitivity analysis, and uncertainty quantification (07 1999)
7. ELLIS: European lab for learning & intelligent systems, https://ellis.eu/, [Online; accessed 22-April-2025]
8. Feinberg, J., Langtangen, H.P.: Chaospy: An open source tool for designing methods of uncertainty quantification. Journal of Computational Science **11**, 46–57 (2015). https://doi.org/https://doi.org/10.1016/j.jocs.2015.08.008, https://www.sciencedirect.com/science/article/pii/S1877750315300119
9. Groen, D., Arabnejad, H., Jancauskas, V., Edeling, W.N., Jansson, F., Richardson, R.A., Lakhlili, J., Veen, L., Bosak, B., Kopta, P., Wright, D.W., Monnier, N., Karlshoefer, P., Suleimenova, D., Sinclair, R., Vassaux, M., Nikishova, A., Bieniek, M., Luk, O.O., Kulczewski, M., Raffin, E., Crommelin, D., Hoenen, O., Coster, D.P., Piontek, T., Coveney, P.V.: Vecmatk: a scalable verification, validation and uncertainty quantification toolkit for scientific simulations. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **379**(2197) (Mar 2021). https://doi.org/10.1098/rsta.2020.0221, http://dx.doi.org/10.1098/rsta.2020.0221
10. Groen, D., Arabnejad, H., Suleimenova, D., Edeling, W., Raffin, E., Xue, Y., Bronik, K., Monnier, N., Coveney, P.V.: Fabsim3: An automation toolkit for verified simulations using high performance computing. Computer Physics Communications **283**, 108596 (2023). https://doi.org/https://doi.org/10.1016/j.cpc.2022.108596, https://www.sciencedirect.com/science/article/pii/S0010465522003150
11. HiDALGO2: HiDALGO2 project webpage. https://hidalgo2.eu (2025), [Online; accessed 22-April-2025]
12. HiDALGO2, CIRCE, SEAVEA: Hackathon on verification, validation, and uncertainty quantification (vvuq) for global challenge applications. Hybrid event organized by HiDALGO2, CIRCE, and SEAVEA projects, held at High-Performance Computing Center Stuttgart (HLRS), Germany, and online (jun 2024), https://www.hidalgo2.eu/hackathon-by-hidalgo2-circe-and-seaveavvuq-projects-5-to-7th-of-june-2024/, held from 5–7 June 2024, focusing on the SEAVEA toolkit (SEAVEAtk) and mUQSA integration; registration details at https://forms.gle/4ytFD4TNEWFcA35u9

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6

13. Kulczewski, M., Bosak, B., Kopta, P., Szeliga, W., Piontek, T.: Fostering uncertainty quantification in global challenges with muqsa toolkit. In: Wyrzykowski, R., Dongarra, J., Deelman, E., Karczewski, K. (eds.) Parallel Processing and Applied Mathematics. pp. 35–46. Springer Nature Switzerland, Cham (2025)
14. mUQSA: mUQSA documentation webpage. https://muqsa.multiscale.pionier-lab.pionier.net.pl (2025), [Online; accessed 22-April-2025]
15. PIONIER-LAB: PIONIER-LAB project webpage. https://pionier-lab.pionier.net.pl (2025), [Online; accessed 22-April-2025]
16. QCG: QCG-PilotJob webpage. https://qcg-pilotjob.readthedocs.io (2025), [Online; accessed 22-April-2025]
17. QCG: QCG webpage. https://qcg.psnc.pl (2025), [Online; accessed 22-April-2025]
18. Richardson, R., Wright, D., Edeling, W., Jancauskas, V., Lakhlili, J., Coveney, P.: Easyvvuq: A library for verification, validation and uncertainty quantification in high performance computing. Journal of Open Research Software **8**(1), 1–8 (Apr 2020). https://doi.org/10.5334/JORS.303
19. SEAVEA: Software environment for actionable vvuq-evaluated exascale applications, https://www.seavea-project.org/, [Online; accessed 22-April-2025]
20. Sullivan, T.J.: Introduction to uncertainty quantification. Texts in Applied Mathematics **63** (2015). https://doi.org/10.1007/978-3-319-23395-6, a beginner-friendly text explaining UQ concepts, including aleatory and epistemic uncertainties, with examples.
21. Tennøe, S., Halnes, G., Einevoll, G.T.: Uncertainpy: A python toolbox for uncertainty quantification and sensitivity analysis in computational neuroscience. Frontiers in Neuroinformatics **12** (2018). https://doi.org/10.3389/fninf.2018.00049, https://www.frontiersin.org/journals/neuroinformatics/articles/10.3389/fninf.2018.00049
22. The American Society of Mechanical Engineers: ASME VVUQ standards. https://www.asme.org/codes-standards/publications-information/verification-validation-uncertainty (2025), [Online; accessed 22-April-2025]
23. VECMA: VECMA project webpage. https://www.vecma.eu (2025), [Online; accessed 22-April-2025]
24. Wikipedia: Compartmental models in epidemiology. https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology (2025), [Online; accessed 22-April-2025]

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97570-7_6