ICCFD13-2026-220

Enabling Reproducible CFD Workflows through Provenance-Aware Execution for an XFOIL-Based Parametric and UQ Study

Steve M. Legensky*, David A. Amels**, Stephen M. Makinen***
Corresponding author: sml@ilight.com
*Intelligent Light, USA
**SmartCFD, USA
***Independent Consultant, USA

ABSTRACT: Modern computational fluid dynamics studies increasingly consist of many related solver executions rather than a single analysis run. Parametric sweeps, surrogate-model training campaigns, grid-refinement evaluations, and post-processing scripts generate large collections of interdependent files whose relationships are difficult to reconstruct after the analysis is complete. This creates a practical barrier to reproducibility, auditability, and reuse, particularly when derived results must be traced back to the solver inputs and intermediate artifacts that produced them.

This paper presents a provenance-aware execution approach for reproducible CFD workflow management. In this approach, a CFD study is represented as an executable workflow composed of solver-run stages, consumed and generated artifacts, and post-processing activities. As the workflow executes, provenance is captured automatically, recording not only the workflow definition but also the actual execution history and artifact relationships. The resulting provenance graph provides a query-enabled record of how each result was produced and supports direct inspection of the files associated with individual runs, stages, and derived products.

The method is demonstrated using two aerodynamic workflows. The first is a drag-polar workflow in which repeated CFD runs are executed over a prescribed angle-of-attack range and consolidated into aerodynamic polar results. The second is a surrogate-assisted uncertainty quantification workflow based on the AIAA UQ Challenge Problem for the NACA 2412 airfoil with flap deflection. Samples generated via CFD over five uncertain input parameters are used to train a convolutional neural-network surrogate. The surrogate is then statistically sampled to provide Sobol sensitivity analysis and CDF generation.

The examples show how provenance-aware execution enables an analyst to trace a plotted quantity of interest, sensitivity result, or uncertainty metric back through the workflow stages that produced it. The same provenance record also supports parameter-space audits that compare surrogate sampling extents with the CFD evidence represented in the graph. This explicit record improves artifact access, supports selective re-execution and debugging, and provides a practical foundation for auditable and reproducible CFD studies without imposing additional manual bookkeeping on the analyst. By treating the executed workflow and its artifact relationships as part of the CFD result, the approach turns provenance from passive documentation into an operational mechanism for reproducibility, audit, and model refinement.

Keywords: CFD, Digital Engineering. Uncertainty Quantification, Knowledge Graphs, Provenance

1. Introduction

Modern computational fluid dynamics (CFD) workflows increasingly require a large number of parameterized simulations to support conceptual design exploration, detailed design feedback, and generation of training data for reduced order or surrogate models. Typical examples include angle-of-attack sweeps, grid refinement studies, and repeated analyses under varying operating conditions. While high-performance computing (HPC) platforms enable these studies to be executed efficiently, ensuring that results remain reproducible, auditable, and reusable over time remains a persistent challenge. These studies include many calculation instances, sampled over a multi-axis domain of interest, that are challenging for analysts to manually manage and the expended effort to do so detracts from the quality focus on the reduced-order/surrogate model or product design.

A *Process-centric CFD Execution Model* is designed to support *Provenance-aware* operation where artifacts utilized and generated in a process that enhances investigation capabilities. This provides for an efficient resolution of runtime errors, surrogate model refinement, and is a foundation for evaluating surrogate model uncertainty, which is still a fertile area for research. Utilization of this type of database framework for this process enables a more amenable procedure for analysts since the data (scripts and simulation data) is controlled, which is a core functionality of data management. The traceability requirements for verification and validation (V&V) can only be adequately addressed through application of provenance concepts, and Government technology development contracts in more recent years are enforcing the requirements in various guidance documents from NASA and DoD [1],[2],[3]. Industry is moving in this direction as well [4].

Non-deterministic evaluation (NDE) studies such as uncertainty quantification (UQ) are a prime example for demonstrating the need to manage a massive number of process generated artifacts. In practice, UQ studies typically include hundreds or thousands of calculation instances, sampled over a multi-axis quantity-of-interest (QoI) domain, to characterize the relationship to system response quantities (SRQs). A portion of these calculation instances are utilized to generate training data for a surrogate model that can be used to fully evaluate, more cost-effectively, the various sources of uncertainty (model inputs, numerical approximations, and model form uncertainty).

The current study addresses the problem of CFD results traceability and reproducibility by treating the set of files utilized in a process and associated relationships between them as a first-class computational object. The management, or data control, of a process's files occurs through a provenance-aware execution method that explicitly records the relationships among simulation parameters, solver runs, generated artifacts, and post-processing steps while a given workflow executes. Furthermore, file access is virtualized, simplifying review of the many file artifacts used and created during workflow execution. The approach is demonstrated using a parametric CFD workflow based on the XFOIL panel method, representative of the repeated analyses commonly performed in preliminary aerodynamic design, surrogate model building and UQ studies. The value of the provenance-aware method can be clearly understood for efficiently working through the use cases of large, parameterized studies described in the next section.

2. Motivating Use Cases

Two applications of CFD in current aerospace and defense programs illustrate the need for effective tracking and reproducibility: aero databases and uncertainty quantification (UQ). Traditionally, many CFD runs are used to either directly characterize vehicle performance under various operating conditions, computing QoIs such as lift, drag, moment and control forces, then storing these tabulated results in database (called a 'loads' or 'aero' database). There may be hundreds or thousands of runs for each design concept or iteration.

UQ applies statistical methods to characterize the 'confidence' that an analyst should have in a computational simulation. Examples such as the ASME V&V 20 [5], and the Roy-Oberkampf Uncertainty framework [6] require extensive simulations, post-processing workflows and management of the simulation results and experimental data to quantify contributions of uncertainty sources (model input, numerical, and model form) to the total uncertainty for any given study or design.

Due to the statistical nature of UQ methods, data-driven surrogate models have gained acceptance as a technique to reduce the amount of actual CFD computation required to supply the many thousands of samples needed to perform UQ [7]. Surrogate models approximate vehicle behavior at lower computational cost than direct CFD computation. In aero databases, CFD computations are performed at varying operating conditions. These results are used to train a mathematical model of the system that can be sampled at much lower computational cost. By using careful sampling methodologies and models that range from kriging to convolutional neural networks, a framework for efficient estimation of QoIs across an entire trade space can be developed.

In both aero databases and surrogate models, executing and reproducing the large-scale CFD run campaign can be challenging for a number of reasons, a few of which are summarized in Table 1:

Many directories and files are usually required for the CFD process	Difficult to navigate thus prone to human error, costs time to create and manage directories and files
Many CFD jobs to submit and check	If runs crash or fail to converge, it costs time to make a determination, repair, and re-submit
Common practice uses ad-hoc scripts and a diversity of file formats	Very difficult to accurately reproduce such a workflow at a later time or audit the veracity of results

Table 1 - Issues that hamper auditing and reproducing large-scale CFD campaigns

Building surrogate models can exhibit further difficulty related to coverage of the desired operating regime: the training domain boundaries may not align to the edges of the design envelope for the product design. This may be due to errors in setting up the CFD sampling process or if it is difficult to obtain converged CFD results for the extreme system response beyond the design envelope, which is a core functionality of UQ for engineering design.

For this research, a drag-polar aero database and a UQ study based on our prior work [8] on the AIAA UQ Challenge Problem [9], were used as demonstration cases. Both of these campaigns used MIT's XFOIL panel method airfoil software [10] as the CFD code. For the workflow process and data management, the IntelliTwin® software from Intelligent Light is used[11].

3. Process-centric CFD Execution Model

The proposed process-centric model, implemented in the IntelliTwin® software, enhances investigation capability to enable users to focus on the theory and design support, remain aligned to broader organization processes, collaborate, and extend technical capabilities. The framework is designed to have a minimal footprint in the work environment to enable engineers to focus on the well-designed software applications with which they are familiar while transparently recording traceability information in real-time thorough execution.

The minimal footprint enables users to follow their own self-optimized processes instead of expecting them to reorganize habits around a new platform. Users can implement re-usable workflows that may include a range of automation levels from batch execution to interactive steps and leverage previously developed scripts (Python, etc.). Workflows are documented in a spreadsheet or JSON file, where the steps, input/output file artifacts and software codes (geometry repair, pre/post-processing, solver, etc.) are chained together to form a detailed representation. The engineer executes workflows through a zero-footprint web-client beside the terminal windows and virtual desktop that are typically used for HPC work. Calculation jobs submitted to the queue are tracked in a table, reflective of the HPC queue, so that the engineer knows when to start follow-on workflows. Behind the scenes, every run executes inside its own Sandbox which is automatically created on the HPC system when the workflow is started. The Sandbox thus mirrors the folder structures engineers typically create by hand—directories for parameter variations, subfolders for solver outputs, and so on. IntelliTwin® formalizes this pattern into a consistent structure across machines and compute environments. Because all artifacts associated with a run are contained within the Sandbox, the system can capture a complete and authoritative record of lineage without imposing extra effort on the user.

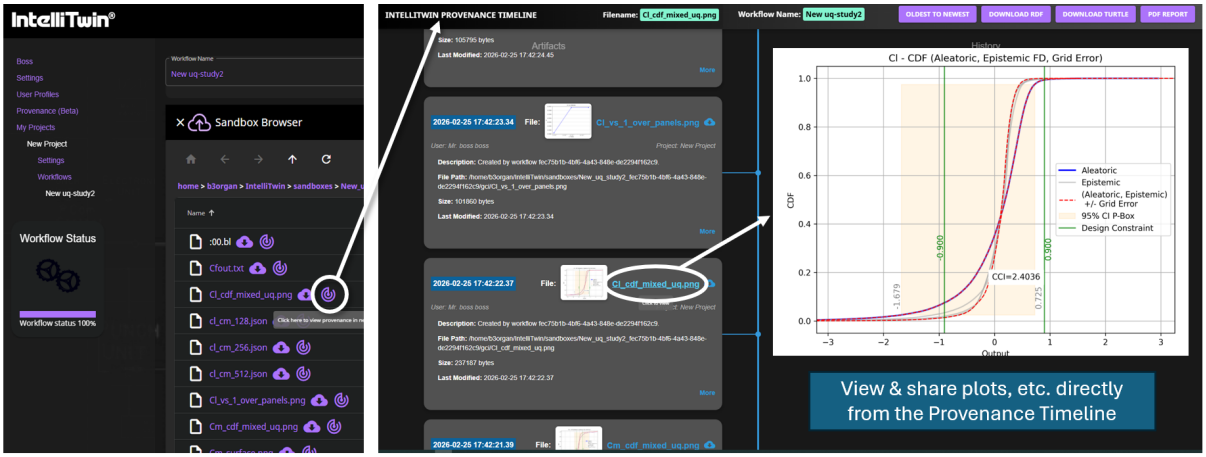

Figure 1: Selecting an artifact from the Sandbox Browser displays its provenance and content

Process steps, job status, artifact metadata is maintained in real time in a lightweight SQL database. A provenance Timeline for any artifact is accessible with a single mouse click in the Sandbox Browser as shown in Figure 1. The Sandbox plus the information in the SQL database form the basis for creating the provenance history. The workflow interface and the Sandbox model strike a balance between usability and the formal demands of digital engineering. Engineers continue to work as they normally do: running analyses, reviewing plots, navigating files, while IntelliTwin® quietly constructs the provenance graph that enables verification, traceability, and long-term reasoning. The provenance, captured into the database, can be utilized

for a range of uses such as identifying a runtime error source (e.g. script) from a known erroneous data file, identifying erroneous results due to compute node failures and non-converged solution instances, and determining the percent complete of a study by querying the database. The provenance capability also enables processes to be re-executed once a scripting error or failed compute node is resolved or for exact reproduction of results (through inversion of provenance to a detailed workflow).

To the authors' knowledge nearly all other SDM/SPDM and PLM platforms do not include workflow provenance, or an equivalent concept, implemented into the database, and the basis for traceability is limited to a defined workflow, which negates retrospective provenance. Also, all other platforms control data through disk read/write operations to and from the storage inside the database domain which leads to very limited scalability for modern HPC workflows. The IntelliTwin® platform is differentiated because it leverages comprehensive provenance concepts matured through the research community and utilizes *virtualized data integration* (i.e. manage/control data in place) and *virtualized data management* (i.e. virtually extend data management domain to any computer system that can run the lowest N-tier architecture process) methodologies to enable massive scalability needed for modern HPC workflows. A more general elaborated description of these concepts, their application, and enabled benefits is included in a previous work by the authors [12].

4. Provenance Capture During Execution

Provenance metadata is collected, transparently to the user, while engineering processes are executed. Provenance can be collected through a variety of capture mechanisms broadly categorized as workflow-based, process-based and operating system-based [13]. The mechanisms are transparent to the user and all can be implemented in a variety of ways. Workflow automation levels can range from full automated to a fully interactive workflow step, with user selected inputs and outputs, to address process steps that cannot yet be automated. The process-based data management approach enables assembly of provenance, in real time, into a database that serves as a knowledge graph that can span over numerous users, computers, and databases that provides a comprehensive description of traceability and is machine auditable.

Table 2 below contains some of the relationships and properties that are used to link and describe artifacts used and produced during workflows. They are derived from the W3C standard PROV-O provenance ontology [14], [15] which forms the basis of the IntelliTwin® ontology. These basic linkages between artifacts, expressed in a knowledge graph provide rich detail on the history of CFD workflows in an open, accessible and most importantly searchable form.

wasGeneratedBy (some activity)	Trace activities in a workflow leading to an artifact
wasAttributedTo (s/w or other agent)	Find software codes / scripts used to produce entity
wasDerivedFrom (preceding artifacts)	Trace all artifacts used to generate this artifact
wasGeneratedAt (time/date of creation)	Establish chronology
value (floating point number or string)	Analysis parameters or Quantity of interest

Table 2: Provenance relationships and properties

As a workflow is run any action that creates, uses or modifies an entity, the provenance of the action is recorded automatically. The relationships in Table 2 appear as 'edges' in the knowledge graph, connecting back from the object that is being acted upon to the activity, software code and predecessor entities from which the current object was derived. A unique identifier is assigned to each entity, its generation time is recorded and any additional annotation such as a numerical value or reference to an external standard is also added. The knowledge graph is a 'directed graph' that can be queried in a variety of ways, and either forward or backward with respect to the edge's direction and it is easy to trace along the edges from a starting value to a final artifact.

Workflow templates, software codes, input decks, results files, and even visualization image files are meticulously cataloged in this way. Referring to Figure 2, the 'cl_vs_alpha.png' file can be traced back via the thick green path ('wasDerivedFrom') all the way back to the Alpha and Mach values used for the CFD calculations. The dashed paths ('wasAttributedTo') lead back to the software codes used to produce the artifacts.

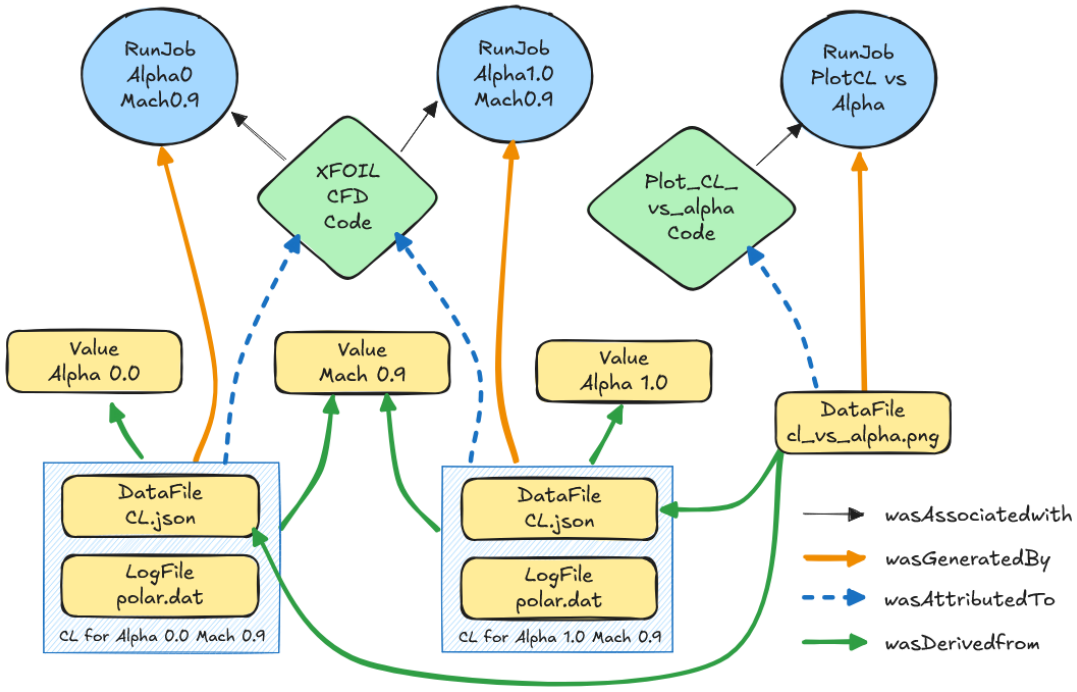

Figure 2: XFOIL-sweep-workflow provenance record; Activities (Blue), Software Codes (Green) & Entities (Yellow)

Thus, the detailed provenance graph represents the signature of a CFD campaign. For UQ, it supports the use of sensitivity characterizations and provides a record of the data and processes used in the study. Furthermore, the knowledge captured in the graph can inform new UQ studies and provides a flexible search through prior analyses and studies.

5. XFOIL-Based Demonstration Workflows

The process-centric methodology is demonstrated using a panel-method aerodynamic analysis implemented with MIT's XFOIL code [10]. An airfoil is selected, along with a fixed set of solver control parameters. IntelliTwin® is used to create individual input decks for the sequence of solution runs. In the case of the drag polar, a simple alpha sweep is performed. In the UQ example,

IntelliTwin® uses Latin Hypercube Sampling (LHS) over a five (5) variable parameter space, as well as solving for three (3) different panel count resolutions in order to compute the Grid Convergence Index (GCI). In all cases the workflows are submitted and monitored via a web-browser interface, with jobs scheduled on a compute resource via PBS.

5.1 Drag-polar / parametric sweep workflow

In the drag polar workflow, the airfoil geometry and Mach number (0.9) are held fixed while the angle of attack is varied over a range of zero to fifteen degrees (0-15°), producing a sequence of solver runs whose results are subsequently consolidated and postprocessed to generate aerodynamic polars.

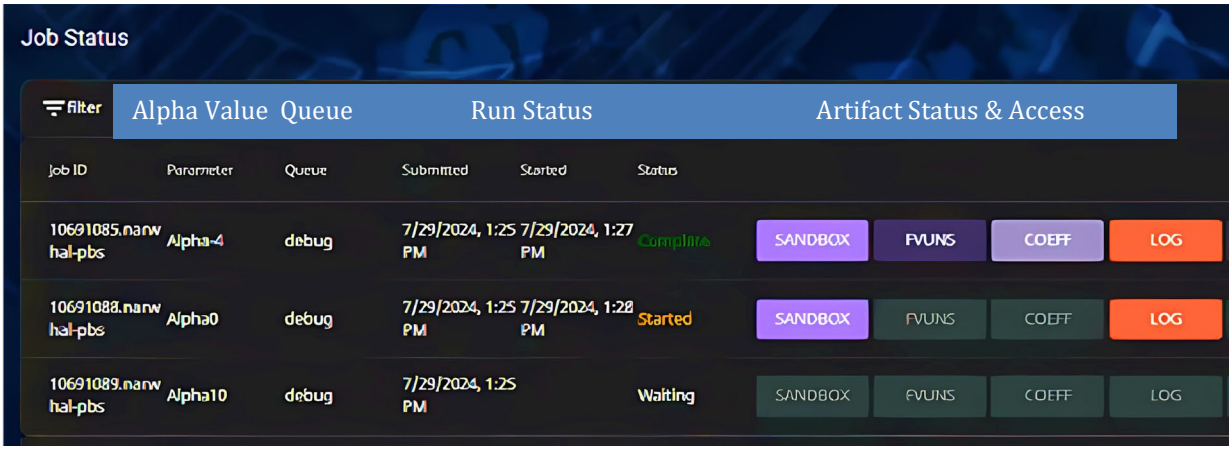

Figure 3: Alpha sweep jobs shown in the IntelliTwin® Job Status dashboard (blue legend box added for clarity)

After setting the fixed Mach value to 0.9 and the range and step size (1°) for the Alpha sweep, the user presses the "Run Workflow" button. IntelliTwin® begins recording the workflow template, run matrix parameters and then prepares all of the scripts for PBS submission. The Job Status dashboard shown in Figure 3 shows the job IDs, queue information, run status and also provides active buttons to review/retrieve artifacts as they become available. In the example above, the first job, "Alpha=4", has completed and all of the artifacts are available. The provenance record is comprehensive and includes not only the run matrix and QoI information, but complete PBS logs for the processes that run remotely.

As shown in the preceding Figure 2, the workflow has two stages: XFOIL-sweep and Plot_Cl. All of the XFOIL-sweep jobs must complete first before the Plot_Cl stage is invoked to gather the individual job results and post-process them into the desired drag polar plot, shown here in Figure 4.

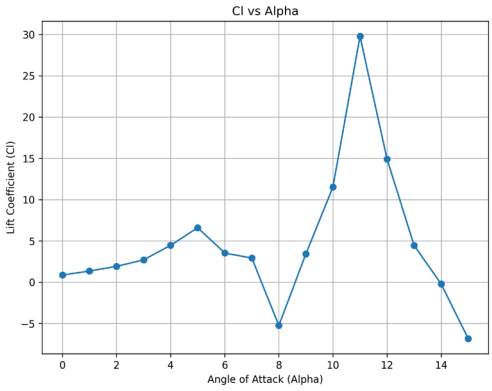

Figure 4: Resulting Cl versus Alpha plot

A simple query as to the stages and artifacts used to create the plot image ('cl_vs_alpha.png') results in a tabular output, shortened to include one XFOIL job for clarity. Table 3 below has two

rows corresponding to the two stages mentioned above. Job names, software name and role is indicated as well as the product for each stage.

stages	jobs	sw	role	products
"Plot_Cl"	"plot_cl_vs_alpha"	"plot_cl_vs_alpha.py"	:Postprocessing	"cl_vs_alpha.png"
"XFOIL-sweep"	"alpha0mach0.9"	"XFOIL"	:CFDCode	"Cl.json"

Table 3: Query result for a single job in the two-stage drag polar workflow in Figure 2

5.2 UQ workflow with surrogate-model branch

The second demonstration workflow extends the repeated XFOIL execution pattern to a surrogate-assisted uncertainty quantification study based on the AIAA UQ Challenge Problem. The workflow uses the NACA 2412 airfoil with flap deflection and the five uncertain input parameters used in the previous ICCFD12 study: angle of attack, Reynolds number, upper transition trip location, lower transition trip location, and flap deflection angle. These parameters define the model-input space for the XFOIL sampling campaign and provide the inputs to the surrogate model.

The workflow is organized as four executable stages: an XFOIL sampling stage ('xfoil-sample'), a surrogate-training stage ('Surrogate_Train'), a surrogate-based Sobol/CDF stage ('Surrogate_Sobol'), and a grid-convergence / mixed-UQ post-processing stage ('GCI'). In the XFOIL sampling stage, IntelliTwin® creates a set of run directories from Latin Hypercube sampled values of the five UQ parameters. In the demonstration workflow, the sampling definition uses 350 LHS samples with a fixed random seed. Each sampled run invokes XFOIL using the corresponding parameter set and produces the solver and post-processing artifacts needed for later stages, including the parameter record, XFOIL output files, coefficient data, pressure-coefficient data, and plots such as 'cp_vs_x.png'.

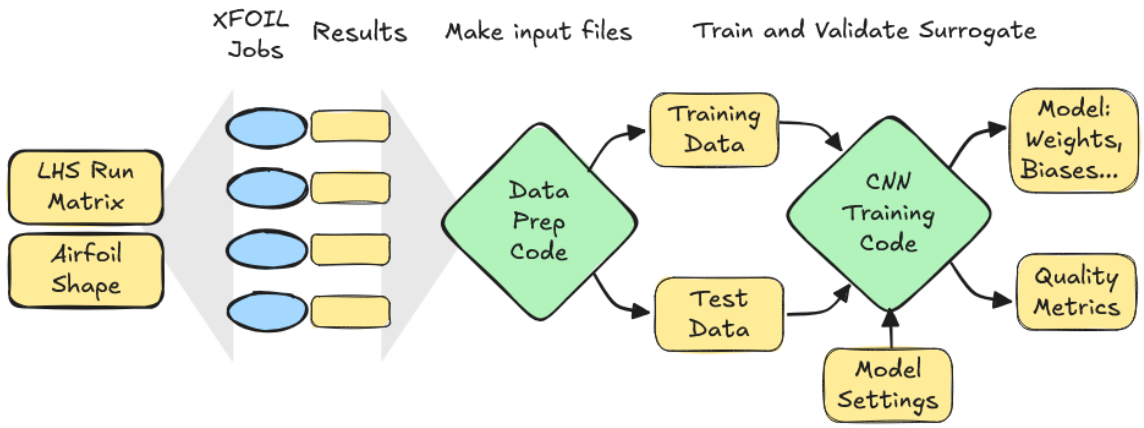

Figure 5: Processes and artifacts in the CNN training workflow

The surrogate-training stage (Figure 5) consumes the XFOIL sampling sandbox and trains a convolutional neural network surrogate model from the solver-input parameters and the resulting quantities of interest. The workflow records both the trained surrogate artifact, 'trained_cnn_surrogate.pkl', and the normalization description, 'mean_std.json', which specifies the mean and standard deviation used for the five input parameters. The training command, input sandbox, output model file, normalization file, and parameter ordering are therefore represented

as workflow artifacts and execution relationships rather than being treated as undocumented side effects of a script.

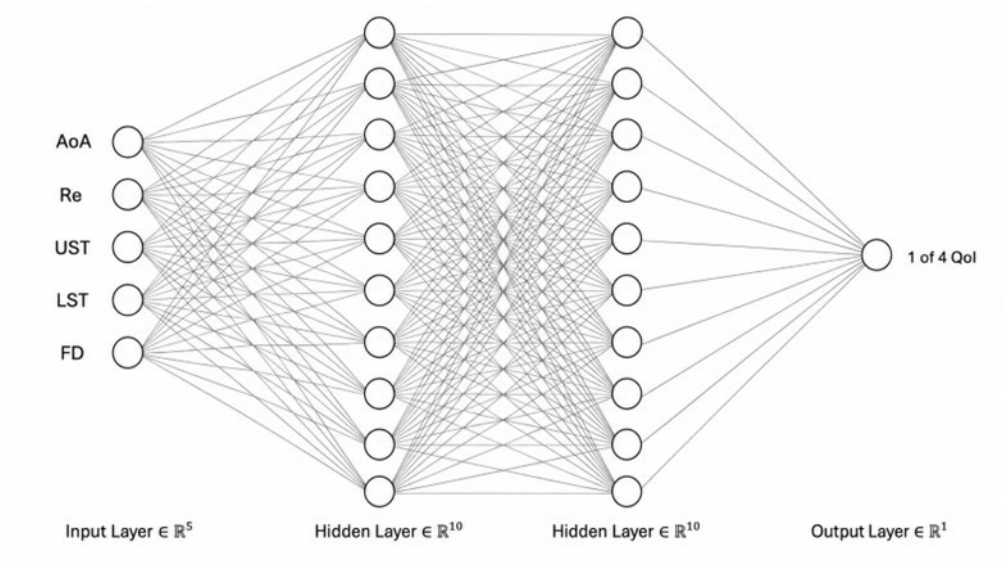

Figure 6: Convolutional Neural Network (CNN) Topology for Surrogate Model

The surrogate Sobol stage then uses the trained CNN surrogate (Figure 6) to perform high-volume statistical sampling at much lower computational cost than direct XFOIL execution. In the current workflow, the surrogate driver is executed with 50,000 samples and produces CDF plots for lift and moment coefficient, a total-order Sobol sensitivity plot, a serialized CDF data object, compressed LHS sample files, and a compact summary of the LHS parameter extents. These outputs are captured as provenance-linked artifacts so that a plotted CDF or sensitivity result can be traced back to the trained surrogate, the normalization description, the training sandbox, and the XFOIL runs used to construct the surrogate.

The final GCI stage adds a numerical-uncertainty component by running XFOIL with multiple panel resolutions and combining the resulting panel-refinement information with the surrogate-generated CDF data. In the present workflow, XFOIL is executed using 512-, 256-, and 128-panel input decks. The resulting coefficient data and convergence plots are used to produce mixed-UQ CDF figures for lift coefficient shown in Figure 7. This stage demonstrates that provenance capture is not limited to solver execution or surrogate training alone; it also covers downstream uncertainty-analysis products that combine solver outputs, surrogate outputs, and post-processing scripts.

Taken together, the four stages provide a compact and representative CFD workflow: sampled solver execution, data-driven surrogate construction, surrogate-based statistical propagation, and numerical-uncertainty post-processing. Each stage produces artifacts that are meaningful to a CFD analyst, while the provenance graph records the relationships among parameter values, XFOIL runs, scripts, trained surrogate files, sensitivity plots, CDF plots, and grid-convergence outputs.

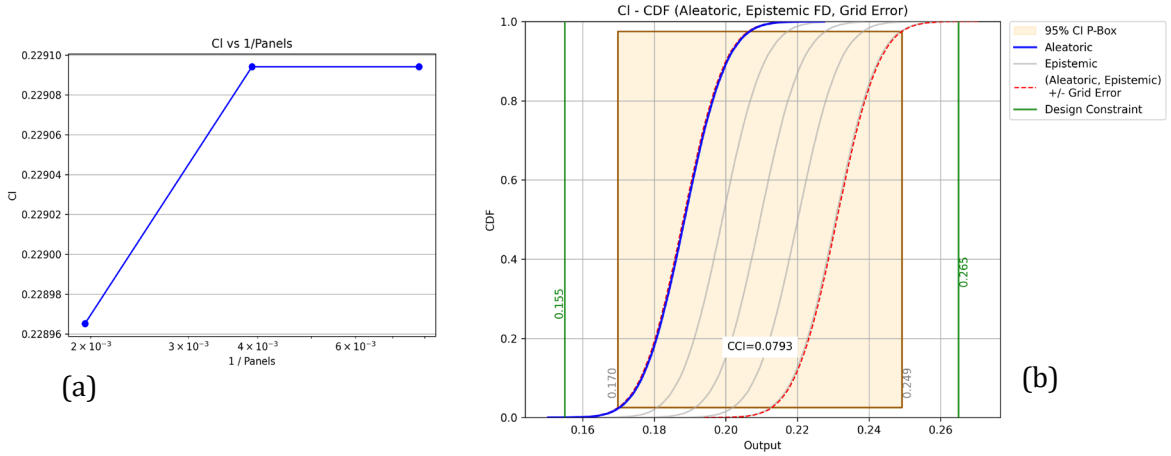

Figure 7: (a) GCI result, 128,256,512 panels; (b) CDF for Uncertainty of Cl, 95% Confidence Interval in shaded box displays a range of 0.170 to 0.249 for Cl, within the Design Constraint

6. Results and Reproducibility Analysis

The XFOIL UQ workflow produces the expected engineering products: coefficient files, pressure-coefficient plots, CDF plots, Sobol sensitivity plots, trained surrogate artifacts, and grid-convergence figures. The provenance record adds the ability to reconstruct how each of those artifacts was generated and how it relates to the parameterized solver campaign.

6.1 Workflow Execution Record

The UQ workflow execution produced a four-stage computational history shown in Table 4. The 'xfoil-sample' stage generated the sampled XFOIL run set from the five-parameter LHS definition. The 'Surrogate_Train' stage consumed that sampling sandbox and generated the CNN surrogate artifact and normalization description. The 'Surrogate_Sobol' stage consumed the trained surrogate and normalization file, executed a 50,000-sample surrogate propagation, and generated the CDF, Sobol, LHS sample, and LHS summary outputs. The 'GCI' stage then combined XFOIL panel-refinement calculations with the surrogate CDF data to produce mixed-UQ and convergence artifacts.

Stage	Execution	Main artifacts	Provenance value
1: xfoil-sample	LHS XFOIL sampling over five UQ parameters; final sample count 350	parameters.json, XFOIL input/output files, coefficient data, pressure-coefficient data, cp_vs_x.png, polar.dat	Links sampled parameters to solver invocations and generated CFD artifacts
2: Surrogate_Train	CNN-style surrogate trained from sampled solver inputs and QoI outputs	trained_cnn_surrogate.pkl, mean_std.json	Captures model artifact, normalization assumptions, parameter ordering, and

			training data lineage
3: Surrogate_Sobol	50,000 surrogate samples for CDF and Sobol analysis	Cl_cdf.png, Cm_cdf.png, sobol_total_order.png, cdf_data.pkl, surrogate_lhs_summary.json	Links UQ results to trained surrogate and sampled parameter-space summary
4: GCI	XFOIL panel-refinement cases using 512-, 256-, and 128-panel decks	mixed-UQ CDF plots, panel-convergence plots, coefficient outputs	Links numerical-uncertainty products to panel studies and surrogate CDF data

Table 4: Workflow execution record for the XFOIL UQ/CNN surrogate study

This stage-level record is important because the UQ result is not a single solver output. It is a derived product built from (1) solver execution, (2) model training, (3) surrogate propagation, and (4) post-processing. Without provenance capture, those relationships must be reconstructed from directory names, scripts, notes, and user memory. With provenance capture, the workflow history is available as a query-enabled record of activities, artifacts, and software invocations.

6.2 Artifact Traceability

A representative trace can begin with a final result artifact such as 'Cl_cdf.png' or 'Cm_cdf.png'. The provenance graph relates that plot to the surrogate activity that generated it, the 'cdf_data.pkl' object used by downstream post-processing, the 'trained_cnn_surrogate.pkl' model used for the predictions, and the 'mean_std.json' file used to normalize the input parameters. Continuing the trace backward reaches the surrogate-training activity, the XFOIL sampling sandbox, and the individual solver-run artifacts used to create the training data. (Recall that this is illustrated in Figure 1 and Figure 2).

The same pattern applies to the grid-convergence products. A mixed-UQ figure such as 'Cl_cdf_mixed_uq.png' is linked to the GCI post-processing activity, the panel-refinement coefficient files, the XFOIL input decks used for the panel calculations, and the surrogate CDF data generated in the previous stage. This provides an auditable chain from final uncertainty figures back to the solver runs and scripts that contributed to them.

This traceability is useful for routine engineering work. If a CDF tail appears unexpected, the analyst can trace backward to determine which surrogate artifact and parameter normalization file were used. If a grid-convergence plot appears inconsistent, the analyst can inspect the specific panel input decks and coefficient outputs used by the GCI stage. If a solver run appears questionable, the analyst can inspect the sampled parameter values and corresponding XFOIL output files without manually searching through a large directory tree.

6.3 CNN Surrogate Signature

The trained CNN surrogate is treated as a workflow artifact with its own provenance signature. That signature includes the model file (weights, biases, etc.), the training command, the input

parameter order, the normalization file, the training sandbox, and the solver-run artifacts used to generate training data. This is important because a surrogate model is not self-explanatory when viewed only as a serialized model file. Its engineering meaning depends on the solver data used to train it, the input parameters assumed by the model, the normalization applied to those parameters, and the quantities of interest used as training targets.

The provenance record therefore supports a more auditable use of the surrogate. Rather than asking only whether a model file exists, the analyst can ask which XFOIL runs contributed to the model, which input parameters were used, which normalization values were applied, which script trained the model, and which downstream CDF or sensitivity artifacts depend on that model. This gives the surrogate model a reviewable computational lineage.

6.4 Parameter-Space Audit and a UQ Radar App

A second benefit is the ability to audit the relationship between the surrogate sampling space and the observed solver-run parameter space. Rapid queries can be used to qualify whether a requested performance points lie within the training space or not. Speed is essential when evaluating tens of thousands of samples. The UQ Radar app, shown in Figure 8, was developed as a lightweight interface for this provenance-derived check. The app queries compact surrogate LHS parameter summaries from the knowledge graph and compares them with observed solver-run parameter ranges extracted from solver-run parameter entities. The comparison is performed for the same five UQ parameters used by the workflow: Alpha, FlapAngle, Reynolds, TripPointUpper, and TripPointLower.

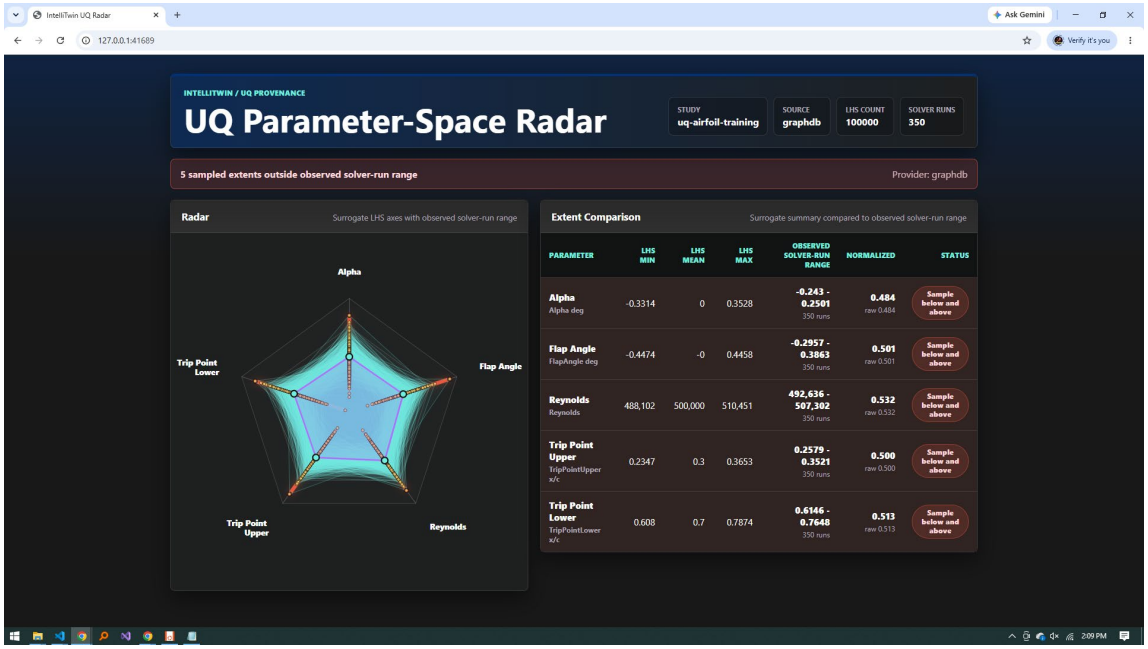

Figure 8: UQ Radar App developed by using the open API to query the provenance graph

For each parameter, the app compares the surrogate LHS minimum and maximum with the observed solver-run minimum and maximum. The comparison identifies whether the surrogate sample extent is inside the observed solver-run range, below it, above it, below and above it, or unknown because no matching solver-run extent was available. The radar chart provides a visual summary of this comparison. Each axis corresponds to one UQ parameter, the axis bounds are

defined by the surrogate LHS summary values, the surrogate mean is plotted as a polygon, and observed solver-run ranges are shown as radial range segments. Segments are highlighted when the surrogate LHS extent extends outside the observed solver-run range.

The radar app is intentionally driven by provenance queries rather than hard-coded spreadsheet values. *Thus, it treats the actual workflow history as the authoritative source of truth.* The surrogate parameter extents are obtained from the knowledge graph entities that record the parameter name, sample count, minimum, maximum, mean, and standard deviation for the surrogate LHS sampling layer. Solver-run ranges are obtained independently from solver-run parameter values. This separation is important because it allows the same comparison to be repeated as new solver runs, surrogate models, or sampling summaries are added to the knowledge graph.

This check should be interpreted as an audit of available provenance, not as a mathematical proof of surrogate accuracy. The observed solver-run range is based on the solver-run parameter entities available in the graph. It therefore provides a practical way to identify possible gaps, setup errors, missing runs, or intentional extrapolation between the surrogate sampling space and the solver-run evidence represented in the provenance database.

7. Discussion and Conclusions

The proposed process-centric CFD execution model has been demonstrated for the AIAA UQ V&V challenge problem to illustrate how it can enhance uncertainty quantification studies in terms of the execution and insight gained. The execution is obviously improved in terms of efficiency and robustness of results by respectively using a well-defined reusable workflow and automatically collecting traceability information. However, the automation also eliminates low level tasks that are not cost-effective for highly skilled/experienced engineers such as manually searching through a hard-drive directory tree and other workflow artifacts to make sense of erroneous results or sensitivities of final results on certain parameters. The provenance timeline feature is an example of this type of task elimination since it improves accessibility of file artifacts to users. Improved insight was demonstrated with the UQ Radar App that shows relationships between the sampling spaces for surrogate model generated results versus the CFD model, which illustrates the merit of final results against utilized training data. The robustness of the study is also improved since the provenance graph stored in the database serves as a unique signature of the surrogate model which would be different if the ordering of the parameters were changed, for example.

The improvements are enabled by the provenance methodology that is at the core of this process-centric approach. Pre-defined workflows are considered prospective provenance because they are forward looking on how history is intended to be created. Workflows support automation and the inherent algorithm describes how data is derived. Retrospective provenance describes history as it actually occurred, which is often different than the original plan. Retrospective provenance refers to utilized workflows, but primarily answers the “what” questions such as what data was derived, by what users, computers, software packages and at what time-window. The combination of prospective and retrospective provenance is a complex web of defined relationships called a provenance knowledge graph. The provenance methodology is implemented with the intention to capture provenance automatically without user action, to avoid dependence on the diligent human actions, and manage (and thereby control) provenance

data in conjunction with engineering data to keep both partitions in sync. The methodology is robustly implemented with programming code and by leveraging the leading provenance ontology [15] to avoid dependence on users' diligence among many other technical tasks.

The process-centric framework enables users to operate at a top-level (e.g. demonstrated four stage UQ workflow) and use automated capabilities to inquire at various levels of detail that are included in the provenance information if necessary for tasks to debug or interrogate results. In this demonstration, provenance was captured for all included artifacts (files, computers, software, etc.) and users ranging from the LHS inputs, XFOIL calculation instances, surrogate model training, and the final UQ study results.

The traceability describes specific instances of files, computers, users, software versions, scripts and workflows. The robust traceability capability provides a foundation to satisfy requirements in documents such as the NASA Standard for Modeling & Simulation and similar documents from other Government agencies, which are more commonly being enforced in recent years [1],[2],[3]. Engineering results can be queried to gain insight, as demonstrated with the UQ Parameter Space Radar app, because algorithms (ranging from SPARQL to AI) can search through the provenance knowledge graph that is managed in sync with the engineering results, which would not be the case for a summary spreadsheet file.

Future engineering industry improvements to reduce design risk, improve product quality, and succeed in previously unachievable challenges will propagate the trend of more physics-based and surrogate model calculations. The larger set of calculations will serve a wider variety of purposes including multiple UQ studies and verification of test data against predictions as digital twin technology matures. The increased number of calculations and design support facets creates a larger challenge for process and data management. Traceability is a critical core capability for the implemented framework because it enables enhancements in the technical work such as debugging workflow processes, rerunning calculations to either recover from runtime errors or demonstrate reproducibility, and compare results referenced by corresponding sets of provenance paths. The methodology also enhances higher-level engineering tasks such as auditing, verification and validation and model refinement as well as other capabilities described in the Future Work section. The implemented provenance and data management capability can be applied to larger CFD models and more complicated workflows just as easily as the minimal data size XFOIL model and workflow utilized in study. With regard to the provenance portion of the method, provenance nodes can figuratively point to large file just as easily as a small file. The implemented virtualized data integration and virtualized data management methods enable management of data in place, which avoids inefficient large file disk read/write operations to move them into and out of a management domain for data control. The platform design was selected to enable the massive data scalability that is needed for HPC engineering processes [12].

8. Future work

A wealth of research has rightly focused on uncertainty sources (model inputs, numerical approximations, and model form uncertainty) relevant to the physics-based simulation model and test data. The overall uncertainty is evaluated through a combination of physics-based simulation results and surrogate model results, predicated on training data from the physics-based simulation and test data. However, the numerical approximation and model form error of the surrogate model should also be evaluated relative to the CFD model, and this is a topic that

has not yet received sufficient attention in the opinion of the authors. Numerical approximation uncertainty of the surrogate model addresses the convergence of the surrogate model results to the CFD results as training data sample distribution and size increases.

The numerical uncertainty in this case would characterize how well the surrogate model captures the input to QoI relationship relative to the CFD model. In addition to the CFD uncertainty sources traditionally considered in total UQ, surrogate-assisted workflows introduce an additional approximation issue: the degree to which the surrogate reproduces the CFD solver response over the sampled parameter domain. This surrogate approximation uncertainty depends on the surrogate family, the density and coverage of the training data, the smoothness of the response, and the training procedure used to construct the surrogate. For example, RBF, CNN, Gaussian-process/Kriging, and PCE surrogates may exhibit different behavior in the interior of the sampled domain than near the edges of the training envelope, particularly when the underlying CFD response is nonlinear or poorly sampled in those regions. In this context, provenance is useful because it records the training samples, solver outputs, surrogate-training artifacts, parameter extents, and post-processing steps needed to audit where a surrogate-based result was interpolated within the training domain, where it approached the sampled bounds, and where additional CFD evaluations may be needed to improve surrogate credibility.

This can be evaluated in terms of different types of surrogate models (e.g. RBF, CNN, PCE) and the order of operations for processing the training data to create the surrogate model (e.g. data shuffling for bias reduction, curriculum learning to enable improved training, and replay buffers to avoid catastrophic forgetting during continual learning). The parameters of the surrogate model can be selected such that the CFD generated results are well captured in the interior of the training data domain. The authors hope to form collaboration partnerships to guide the correct development of capabilities, from a statistics theory perspective, to quantify uncertainty associated with a surrogate model and the contribution of this subset of sources to the total uncertainty.

The process-based data management approach utilized in this study enables a solution approach for evaluation of the numerical approximations of the surrogate model creation and utilization. The captured provenance enables exact reproduction of the SRQs from the discrete set of QoI samples, for training the surrogate model, without repeating the expensive CFD calculations. The provenance graph can be queried to reproduce SRQs, in the neighborhood of interest, of the CFD model for this type of investigation similar to the provenance-enabled explainable AI methodology developed by Zhang et al. [16]. A forward-trace from QoIs to SRQs in the provenance graph would show how QoI inputs contribute to SRQs. A backward trace from a subset of SRQs to QoIs would show the inverse relationship without computational dependence on the SRQs excluded for the sub-study. The authors intend to demonstrate, in future work, a capability to show explainability of the surrogate model as described above.

In recent years, government customers are enforcing requirements from modeling and simulation standards for contract deliverables associated with development of advanced hardware technology. The NASA Modeling & Simulation Standard provides requirements for how this type of data should be included in work products[2]. The standard was recently revised (2024), but the original version was written in 2006. Similar documents from other Government agencies precede this particular NASA standard [1], [3]. The NASA standard, for example, includes 63 instances of the phrase “A record of” which illustrates the importance of documentation and

traceability for the customer to believe the contractor's analysis and design. As is described below, provenance is a well-developed set of concepts that is an optimal solution to the problem statement "A record of" that is intentionally worded to not prescribe how this is accomplished. Compliance with these traceability requirements is particularly difficult for UQ studies that typically include thousands of calculation instances. Current and recent work has developed Systems Engineering ontologies that can be used to describe modeling and simulation requirements for example by Gregory & Salado at Arizona State University [17]. The authors intend to incorporate result of this ontology development work into the demonstrated framework, and demonstrate how users can consistently generate traceable results that can be verified and validated with the customer without human debate about the correct implementation of written prose.

Acknowledgements

The authors would like to express their gratitude to Dr. Earl P.N. Duque, who while at Intelligent Light, lead activities related to our UQ research and software offerings. This includes leading development of SpectreUQ, participating in various AIAA and DOE activities related to Non-deterministic Engineering and UQ. Dr. Duque also worked with the AIAA team to develop the Uncertainty Challenge and was responsible for Intelligent Light's participation in that activity and presentation of results at ICCFD12 in Kobe, Japan [8], [9].

Thank you to Barry Smith and John Beverly of NCOR at the University of Buffalo (https://ncor.us/) and our membership in OAGi (https://oagi.org) for inspiration and technical guidance related to ontology development.

This work was conducted using Protégé (https://protege.stanford.edu/about/) & MIT's XFOIL, available here: https://web.mit.edu/drela/Public/web/xfoil/.

References:

[1] "ARMY - DA PAM 5-11 - VERIFICATION, VALIDATION, AND ACCREDITATION OF ARMY MODELS AND SIMULATIONS | GlobalSpec." Accessed: Jun. 11, 2026. [Online]. Available: https://standards.globalspec.com/std/481733/da-pam-5-11

[2] J. W. Pellicciotti, *NASA-STD-7009B-Final-3-5-2024*, 7009B, Mar. 05, 2024. Accessed: Jun. 01, 2024. [Online]. Available: https://standards.nasa.gov/sites/default/files/standards/NASA/B/1/NASA-STD-7009B-Final-3-5-2024.pdf

[3] *DOCUMENTATION OF VERIFICATION, VALIDATION, AND ACCREDITATION (VV&A) FOR MODELS AND SIMULATIONS*, MIL-STD-3022, Jan. 28, 2008. Accessed: Jun. 01, 2026. [Online]. Available: http://www.everyspec.com on 2012-05-22T15:09:15.

[4] R. Eshcol *et al.*, "A Digital Engineering Quality Assurance Framework for Engineering Models and Simulation Data," in *AIAA SCITECH 2026 Forum*, 0 vols., in AIAA SciTech Forum. , American Institute of Aeronautics and Astronautics, 2026. doi: 10.2514/6.2026-2121.

[5] American Society of Mechanical Engineers, Ed., *Standard for verification and validation in computational fluid dynamics and heat transfer: an American national standard*, Reaffirmed 2016. in ASME V&V, no. 20–2009. New York, NY: The American Society of Mechanical Engineers, 2009.

[6] C. J. Roy and W. L. Oberkampf, "A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing," *Comput. Methods Appl. Mech. Eng.*, vol. 200, no. 25–28, pp. 2131–2144, Jun. 2011, doi: 10.1016/j.cma.2011.03.016.

[7] A. L. Kaminsky *et al.*, "Design and Application of the Sage Surrogate Modeling Software," in *AIAA SCITECH 2025 Forum*, 0 vols., in AIAA SciTech Forum. , American Institute of Aeronautics and Astronautics, 2025. doi: 10.2514/6.2025-0037.

[8] E. P. N. Duque, M. D. Burklund, D. A. Amels, and S. M. Legensky, "Total Uncertainty Quantification of the AIAA UQ Challenge Problem using an Interactive Web Browser Based Tool for Data and Automated Workflow Management," *Comput. Fluid Dyn.*, 2024.

[9] A. W. Cary, J. A. Schaefer, E. C. DeCarlo, E. P. Duque, and M. Khurana, "Overview of Fluid Dynamics Uncertainty Quantification Challenge Problem and Results," in *AIAA SCITECH 2024 Forum*, Orlando, FL: American Institute of Aeronautics and Astronautics, Jan. 2024. doi: 10.2514/6.2024-0705.

[10] M. Drela, "XFOIL: An Analysis and Design System for Low Reynolds Number Airfoils," in *Low Reynolds Number Aerodynamics*, vol. 54, T. J. Mueller, Ed., in Lecture Notes in Engineering, vol. 54. , Berlin, Heidelberg: Springer Berlin Heidelberg, 1989, pp. 1–12. doi: 10.1007/978-3-642-84010-4_1.

[11] Intelligent Light, *IntelliTwin*®. Intelligent Light. [Online]. Available: https://www.ilight.com

[12] S. M. Legensky, D. Amels, and S. M. Makinen, "Enabling Scalable Provenance-Based Data Management for Digital Engineering Workflows: A Semantically Grounded Approach for HPC Scale CFD," in *AIAA SCITECH 2026 Forum*, 0 vols., in AIAA SciTech Forum. , American Institute of Aeronautics and Astronautics, 2026. doi: 10.2514/6.2026-1529.

[13] J. Freire, D. Koop, E. Santos, and C. Silva, "Provenance for Computational Tasks: A Survey," *Comput. Sci. Eng.*, vol. 10, no. 3, pp. 11–21, May 2008, doi: 10.1109/MCSE.2008.79.

[14] L. Moreau, P. Groth, J. Cheney, T. Lebo, and S. Miles, "The rationale of PROV," *J. Web Semant.*, vol. 35, pp. 235–257, Dec. 2015, doi: 10.1016/j.websem.2015.04.001.

[15] "PROV-O: The PROV Ontology." Accessed: Jun. 11, 2026. [Online]. Available: https://www.w3.org/TR/prov-o/

[16] J. Zhang, W. Zhou, and B. E. Ujcich, "Provenance-Enabled Explainable AI," *Proc. ACM Manag. Data*, vol. 2, no. 6, pp. 1–27, Dec. 2024, doi: 10.1145/3698826.

[17] J. Gregory and A. Salado, "TOWARDS A SYSTEMS ENGINEERING ONTOLOGY STACK," *INCOSE Int. Symp.*, vol. 34, no. 1, pp. 1304–1318, Jul. 2024, doi: 10.1002/iis2.13210.