Software Layout Plan for a unique AI Framework

Table of Contents

List of Figures

List of Tables

Executive Summary

The key objective of this Deliverable is to provide an initial description of a software layout plan for a unique Articial Intelligence (AI) framework enabling AI & High-Performance Computing (HPC) methods at Exascale. Co-designed by the different Work Package (WP) use cases of WP3 and WP4 of the Center of Excellence in Exascale Computing “Research on AI- and Simulation-Based Engineering at Exsascale” (CoE RAISE), the objective of a reference architecture of this framework is to be generally applicable to those use cases and also to other use cases beyond RAISE. Hence, the objective of this Deliverable is to propose a reference architecture that is reasonably specific to describe a scalability-proven set of AI packages and harmonized with the existing versatile HPC hardware infrastructures of RAISE.

The result of this Deliverable is an initial list of AI framework requirements identified by the WP3 and WP4 use cases, taking opportunities and constraints from current HPC software and hardware infrastructures into account. The use case applications have been particularly selected in the project, because they require Exascale resources raising requirements for a unique AI framework that offers high scalability. The tangible results and outputs of this WP2 Deliverable are as follows: (i) provide kernels for Jupyter notebooks with correct version setups of modules for specific HPC systems; (ii) provide a lightweight and abstract Python Application Programming Interface (API) building on Open Neural Network Exchange (ONNX) enabling easy exchange with MLFlow addressing; (iii) provide a lightweight Python API that abstracts from low-level versioning of AI packages (with proven scalability) and that is harmonized with different available HPC system module versions; (iv) provide containers in Singularity with prepackaged datasets and required software stacks needed for AI models.

1 Introduction

One of the objectives of the Work Package (WP) “AI- and HPC-Cross Methods at Exascale” (WP2) activities in the Center of Excellence in Exascale Computing “Research on AI- and Simulation-Based Engineering at Exsascale” (CoE RAISE) is designing, developing, and provisioning a unique Artificial Intelligence (AI) framework that enables AI at Exascale. In addition, the WPs “Compute-Driven Use-Cases towards Exascale” (WP3) and “Data-Driven Use-Cases towards Exascale” (WP4) of RAISE work on the convergence of traditional High-Performance Computing (HPC) and innovative AI techniques driven by different use cases. 'AI at Exascale' in RAISE means to develop unique AI framework methodologies that are co-designed by these use cases and usable by a wide variety of scientific and engineering applications in the Exascale era.

The framework needs to address a broad range of HPC use case workloads intertwined with complementary AI training and inference using advanced AI algorithms. Using data augmentation techniques and pre-trained neural networks through transfer learning (TL), the RAISE AI framework enables new approaches for scientific and engineering applications while saving valuable computational processing time for computations at Exascale. Based on open-source community toolsets and HPC best practices for AI, the framework will provide methodologies to scale up AI on a large number of GPUs to keep pace with the ever-increasing number of applications that require Exascale computing. To tackle the fundamental problem of using AI in practical applications, with respect to hyper-parameter tuning requirements, the RAISE framework will leverage cutting-edge Neural Architecture Search (NAS) techniques driven by Reinforcement Learning (RL) and evolutionary computation methods.

The purpose of this document is to develop the first blueprint of a software layout plan for the AI framework, including the significant requirement analysis outputs obtained from the ‘Interaction Room’ process in WP2. It also gives the reader an impression of the complexity of designing and developing such a framework that needs to interact with many innovative technologies. Therefore, the purpose of this Month 9 Deliverable is to provide readers with an initial version of the software layout plan as a basis for continuous updates in a living document.

The structure of this Deliverable is as follows. Section 2 offers a short background of relevant technologies to better understand the co-design use case application requirements and the derived software layout plan. Section 3 then reflects on the ‘Interaction Room’ process and lists a clear set of requirements obtained from the use cases. The initial version of the software layout plan for a unique AI framework is then presented in Section 4.

This document is primarily intended for the WP3 and WP4 use case application experts of RAISE. It provides thus initial information for potential adopters of the unique AI framework by the larger HPC community and RAISE stakeholders.

2 Technology background & challenges

This section provides a brief background of the major technologies involved in the AI framework and its larger environment, such as HPC systems and relevant standards. As this Deliverable focuses on the framework, official sources are referenced as much as possible without losing sight of the essential elements of each topic.

2.1 HPC system environments & RAISE use cases

The primary HPC systems and their environments used in the co-design efforts of the framework are available in the D2.1 [1] and D2.5 [2] Deliverable documents of RAISE. As shown in Figure 1, these environments represent an unprecedented Exascale *hardware infrastructure* (Figure 1, item 1). Based on this solid foundation, a seamlessly usable and versatile *software infrastructure* (Figure 1, item 2) is critical for accelerating convergence through new AI tool sets that are ready to scale for enormous quantities of data.

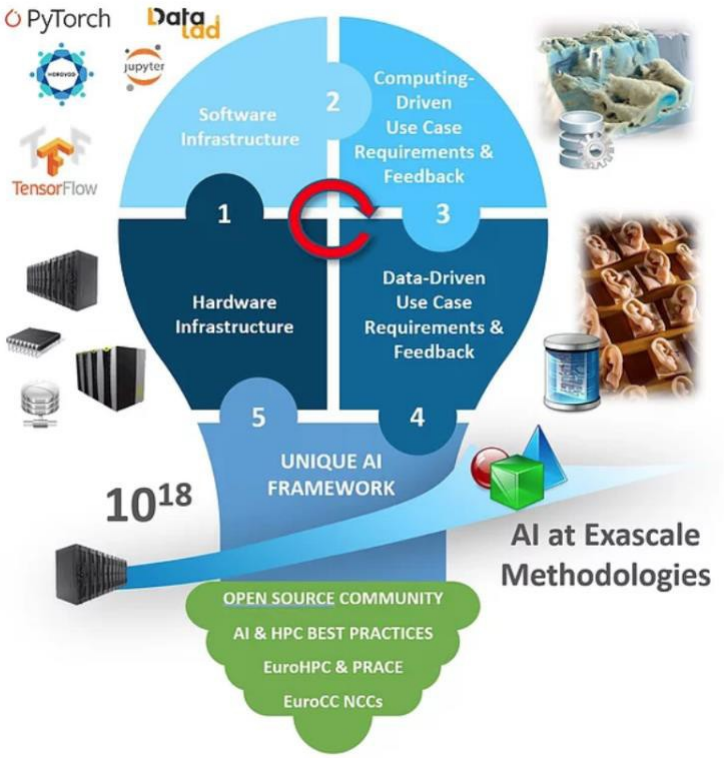

Figure 1: Overview of relevant HPC ecosystem factors influencing the framework layout.

RAISE is performing application co-design based on a variety of use cases[1]. It considers AI requirements of *compute-driven use cases* (Figure 1, item 3) using numerical methods based on known physical laws. RAISE also addresses AI requirements of *data-driven use cases* (Figure 1, item 4) with large measurement device datasets and focuses on AI methods. All these four main key factors and their requirements shape the design of the *unique AI framework* (Figure 1, item 5) to be usable by many other use case applications outside the RAISE consortium.

[1] CoE RAISE Web page Use Cases: https://www.coe-raise.eu/use-cases

2.2 Relevant AI technologies

A wide variety of AI technologies influence the unique AI framework, while RAISE emphasizes those relevant for Exascale and can leverage HPC system environments. In the following, these technologies are briefly described with a focus on their unique selling propositions for RAISE.

A general observation in RAISE in particular and in the AI community is that Python is the preferred language for Machine Learning (ML) and Deep Learning (DL) using AI technologies. It also works seamlessly with C and C++ code modules that are relevant in RAISE when intertwining AI methods with traditional simulation science codes often written in these languages.

From a license perspective, many of the AI technologies are free and open-source, being community-friendly and guaranteeing improvements in the long run. In addition, already existing libraries that often enable solutions for given AI problems are observed.

2.2.1 Basic AI frameworks, tools, and libraries

All use cases in RAISE use innovative approaches of DL networks. Therefore, frameworks, tools, and libraries supporting the use cases are of primary interest to RAISE. One of the most popular open-source libraries is TensorFlow[2] (originally developed by Google) that enables the development and training of DL models. More recently, it also comes directly packaged with the intuitive high-level Application Programming Interface (API) called Keras[3] that enables easy model building, rapid prototyping, fast model iteration, and easy debugging. TensorFlow (with Keras) is available on almost all HPC machines in the community and is adopted in many commercial cloud platforms today.

An increasing uptake of the PyTorch DL library[4] for Python (originally developed by Facebook) is observed in RAISE. For RAISE, the relevance lies in its strong support for C++, e.g., through C++ interfaces. This library is also customizable to develop and train a wide variety of DL models, and is already widely used in the HPC community.

One possible alternative of a DL framework that is increasingly used especially in Germany is the Helmholtz Analytics Toolkit (HeAT)[5]. It is also open-source software optimized for High-Performance Data Analytics (HPDA), ML, and DL. It provides highly optimized algorithms and data structures for creating DL models on HPC systems. WP2 in RAISE organized a seminar with HeAT developers. A recording of this seminar is available through RAISE's YouTube Channel[6].

There are many other basic DL frameworks, tools, and libraries available such as Theano[7] (recently renamed to Aesara because Theano development stopped) or the more broadly used MXNet[8]. As both are currently not used in any RAISE co-design applications, they are not influencing the design of the RAISE AI framework.

[2] TensorFlow Framework: https://www.tensorflow.org/
[3] Keras High-Level Python Deep Learning API: https://keras.io/
[4] PyTorch Deep Learning Library: https://pytorch.org/
[5] HeAT: http://www.helmholtz-analytics.de/helmholtz_analytics/EN/GenericMethods/HeAT/_node.html
[6] CoE RAISE YouTube channel: https://www.youtube.com/channel/UCAdIZ-v6cWwGdapwYxdN7dg
[7] Theano / Aesara: https://github.com/aesara-devs/aesara
[8] MXNet Library for Deep Learning: https://mxnet.incubator.apache.org/versions/1.8.0/

Finally, it is clear that data sciences at large and using AI frameworks such as those listed above also make indirect use of several well-known Python libraries such as Scikit-learn[9], Pandas[10], the Natural Language Toolkit (NLTK)[11], and NumPy[12]. As RAISE primarily focuses on HPC environments, Apache Spark MLLib[13] and other High-Throughput Computing (HTC) and cloud computing technologies are kept out of consideration in the initial blueprint. They may, however, be considered in updates of the initial software layout plan.

2.2.2 Distributed Deep Learning towards Exascale

One of the key concerns in WP2 of RAISE is the scaling-up of distributed DL techniques, which is necessary due to RAISE's continuously increasing volume and complexity requirements demanding Exascale resources. In addition, hyper-parameter optimization and AutoML techniques execute the same basic models with different hyper-parameter configurations that add to the demand for computational Exascale resources. A thorough overview of low-level techniques leveraging parallel and scalable methods on HPC systems is given by Tal Ben-Nun and Torsten Hoefler in [3].

An open-source library for distributed DL for RAISE is Horovod [4] (originally developed by Uber for TensorFlow), which abstracts the complex TensorFlow functionalities. It thus supports other basic DL libraries mentioned above such as PyTorch or MXNet. Horovod offers quite good usability in the sense that adding/changing just a few lines of code to enable the scale-up of DL models. Horovod is often installed on European HPC systems and is used by many scientific communities to inlucde multi-GPU computing in parallel DL model training.

For RAISE, the PyTorch Distributed Data Parallel (DDP) module [5] that is a built-in approach to parallelize the training of DL models across multiple GPUs is also of interest. In other words, PyTorch does not require explicitly written lines of code for scaling-up like Horovod does. This framework is also very popular, given its built-in feature for scaling-up among many scientific and engineering communities.

Finally, the NVIDIA Data Loading Library (DALI)[14] is an open-source framework to accelerate the data loading process in DL applications by involving GPUs. This library also transparently scales across multiple GPUs and can have a significant performance impact on HPC application workloads in RAISE.

2.2.3 Tools supporting the AI model development process

Among the frequently used tools in the AI development cycle are Jupyter notebooks[15], which are also broadly deployed on European HPC systems, e.g., at the Jülich Supercomputing Centre (JSC) in Germany via Jupyter-JSC[16]. Their inherent design is based on JupyterLab, a web-based interactive development environment for Jupyter notebooks. Codes can be developed in different programming languages, and it is well suited for handling datasets. The tool's popularity in the AI and HPC communities was further increased by Google's

[9] Scikit-learn library: https://scikit-learn.org
[10] Pandas library: https://pandas.pydata.org/
[11] Natural Language Toolkit (NLTK): https://www.nltk.org/
[12] NumPy: https://numpy.org/
[13] Apache Spark MLLib: https://spark.apache.org/mllib/
[14] NVIDIA Data Loading Library: https://developer.nvidia.com/dali
[15] Jupyter Notebooks: https://jupyter.org/
[16] Jupyter-JSC: https://jupyter-jsc.fz-juelich.de

Colaboratory ('Google Colab') offering[17] hosting Juypter notebooks and providing GPUs for free. In comparison to RAISE, Google Colab enables no scaling across multi-GPU usage while this is a primary concern for RAISE. The general access and usability of Jupyter notebooks on HPC systems have, however, been repeatedly mentioned as highly important by many RAISE use cases.

Another relevant technology for the ML lifecycle is the MLFlow open-source platform[18]. It enables 'tracking' to record and query DL experiments (i.e., code, data, configurations, and results). It enables 'reproduction' through packaging data science code in a format to run on any platform that supports the variety of different HPC systems used in RAISE. Its 'registration' features enable to store, annotate, discover, and manage AI models in central repositories. These are all features required by RAISE as well. Similiar to MLFlow is the ClearML open-source platform[19]. There is experience in the consortium that the technology is slightly more powerful than MLFlow. It is already used by the RAISE partner Flanders Make (FM), it is open-source, and RAISE investigated its features and deemed it a reasonable technology (only selected security features like user groups are excluded in open-source versions).

2.3 Relevant AI de-facto standards and community sharing

Standards in the area of AI are still relatively new or represent de-facto standards of specific scientific or engineering communities. There are, however, several directions that are becoming more broadly accepted and adopted. These standard topics in AI often refer to sharing of AI models, data, and techniques across different communities. This section of the Deliverable lists some directions of standards and sharing possibilities relevant to RAISE.

2.3.1 Open Neural Network Exchange

The Open Neural Network Exchange (ONNX)[20] is an open standard format built to represent ML and DL models. It is of interest to RAISE in multiple ways: It enables not only sharing of models within the developer teams of each of RAISE's use cases, but also enables sharing across different user communities and across use cases. The reasoning is that some pretrained models or specific DL models can be useful also outside the original scientific domain where they were trained, e.g., when TL beomes possible. ONNX is thus relevant for the unique AI framework in RAISE. It defines a common set of operators that are the building blocks of ML and DL models. In addition, it offers a common file format for DL models to enable AI developers to use models with a variety of frameworks, tools, and runtimes that the RAISE AI framework entails.

2.3.2 OpenML Project

The OpenML project[21] represents a popular movement to build an open, organized, and online ecosystem for ML. In other words, it builds open-source tools to discover and share open data from different application domains to be used in specific ML environments. Data scientists using OpenML can quickly build models together with thousands of other data scientists, analyze their results against the state of the art and results of others, and even receive

[17] Google Colaboratory: https://colab.research.google.com/notebooks/intro.ipynb
[18] MLFlow Platform for the machine learning lifecycle: https://mlflow.org/
[19] ClearML Machine Learning Operations Stack: https://clear.ml/
[20] Open Neural Network Exchange: https://onnx.ai/
[21] OpenML project: https://www.openml.org/

automatic advice on how to build better AI models. All these topics are also relevant in RAISE as many of the same DL models and algorithms are adopted in different use cases.

2.3.3 Containers

Container runtimes like Docker[22] and Singularity [6] are becoming increasingly popular in the AI and HPC community and are increasingly supported on large HPC systems in Europe, e.g., JSC supports containers on its Jülich Wizard for European Leadership Science (JUWELS) system[23]. Containers provide the ability to build, ship, and run scientific applications with all their dependencies. They encapsulate containers from other software and the underlying operating system, and are more lightweight than virtual machines with respect to memory and disk consumption, and CPU overhead.

2.3.4 Conda

Conda[24] is a relevant technology for RAISE as it is a widely used open-source package management system and environment management system used in HPC environments, e.g, it runs on the HPC system Rudens at the Riga Technical University (RTU). Its benefits rely in quick installations, runs, and updates of numerous important python packages for AI and all their dependencies. It easily creates, saves, loads, and switches between environments. The RAISE partner RTU has good experiences in deploying and using it.

2.4 ATOS AI4Sim library

The AI4SIM team of ATOS in the RAISE consortium is designing and building a library that is important for RAISE to consider in its efforts of creating the unique AI framework[25]. It considers the whole AI lifecycle and emphasizes container usage to create products from different AI use cases such as in RAISE. The AI4SIM library is complementary to the efforts in RAISE and could enable sustainability options for the unique AI framework in RAISE being taken on-board of ATOS in their products. Updates on this will be considered in the next version of the layout plan.

For ATOS it is essential to connect the activities in RAISE with the AI4Sim library and thus the AI framework has to be at least compatible to those activities to be used in commercial settings. This in turn raises questions about Intellectual Property (IP) rights and licenses. It should be noted that in RAISE there exist processes that deal with the question on how to integrate libraries from a legal point of view outside of WP2.

[22] Docker Container System: https://www.docker.com/
[23] JUWELS Container Environment: https://apps.fz-juelich.de/jsc/hps/juwels/container-runtime.html
[24] Conda package management and environment system: https://docs.conda.io/en/latest/
[25] Note that there is no public information on the library available at present.

3 Application co-design requirements & benchmark activities

This section informs about the application co-design process, the identified use case requirements, and initial benchmark activities.

3.1 Use case co-design process overview

The co-design activities in WP2 started with a factsheet development process teaming up with WP3 and WP4 use case members. Beyond cross-WP team building, the results have been factsheet drafts for each use case to understand the overall setup and need of software packages and hardware infrastructures. One example of a factsheet is illustrated in Figure 2 while others can be found on the RAISE's Basic Support for Cooperative Work (BSCW) server[26]. The fact sheets are in the process of being finalized until Month 12 of the project. It is planned to use them in use case and project presentations, exploitation, and dissemination activities, and in the mid-term review of the RAISE project as a general overview per use case.

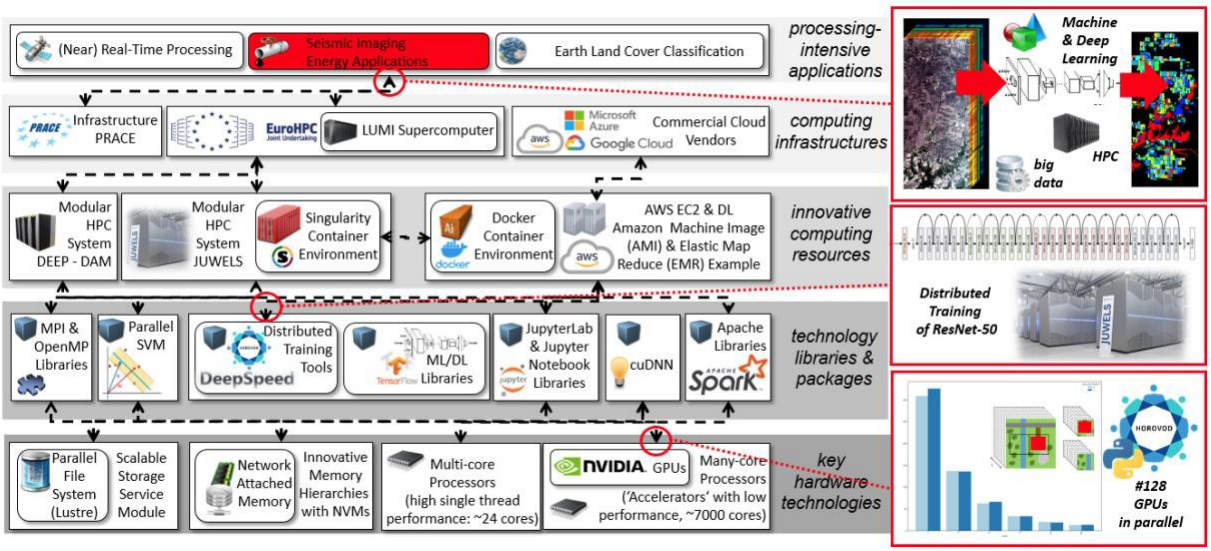

Figure 2: Factsheet of Task 4.3 Seismic imaging with remote sensing [6].

Based on the initial teamwork of building factsheets together, the software engineering process of so-called 'Interaction Rooms'[27] was started with each of the different use cases. WP2 also organized a public RAISE seminar 'HPC Systems Engineering in the Interaction Room' that is available on the RAISE YouTube channel[28]. The CoE RAISE use case teams together with WP2 experts perform co-design activities with this technique. The goal is to understand the domain requirements and technical restrictions and to identify aspects of particular scientific value and the most critical risks. The Interaction Room also outlines initial lessons learned in intertwined HPC and AI use case applications. Using the Interaction Room helps to prevent misunderstandings in early RAISE project stages and can support communication between stakeholders throughout the course of the project when developing the unique AI framework together. Due to the ongoing Covid-19 pandemic, the Interaction Room technique makes use of 'virtual whiteboards' based on Mural Boards[29]. Those boards have been adopting the Interaction Room main methodology and have been specifically tuned with four different

[26] RAISE factsheets: https://bscw.zam.kfa-juelich.de/bscw/bscw.cgi/3340953
[27] CoE RAISE News Items 'Interaction Room Seminar': https://www.coe-raise.eu/news-2021-04
[28] CoE RAISE YouTube Channel, RAISE Seminar Recordings 'HPC Systems Engineering in the Interaction Room': https://www.youtube.com/watch?v=lLkQGPA9_CI
[29] Mural Boards - Virtual whiteboards: https://www.mural.co/

Canvas areas to carve out more detailed requirements from the WP3 and WP4 use cases. One example of a Mural Board is provided in Figure 3, Figure 4, Figure 5, and Figure 6 while a complete list of all Mural Boards is available on the BSCW server[30]. Table 1 provides a brief overview of the four different Canvas areas and how they are used as a communication mechanism across RAISE WPs.

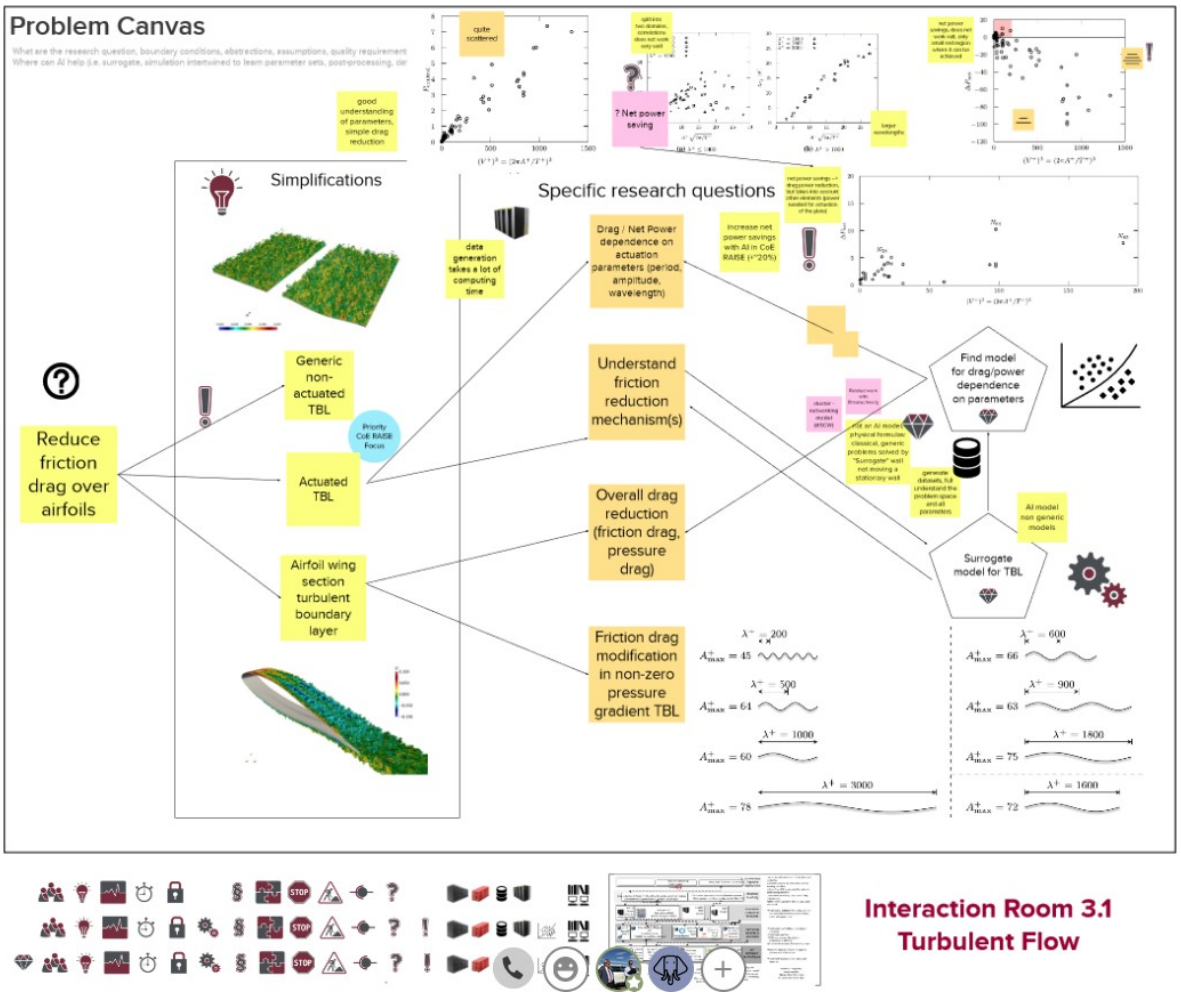

Figure 3: Mural board example of WP3 Task 3.1 – Problem Canvas

[30] RAISE Mural Boards: https://bscw.zam.kfa-juelich.de/bscw/bscw.cgi/3591551

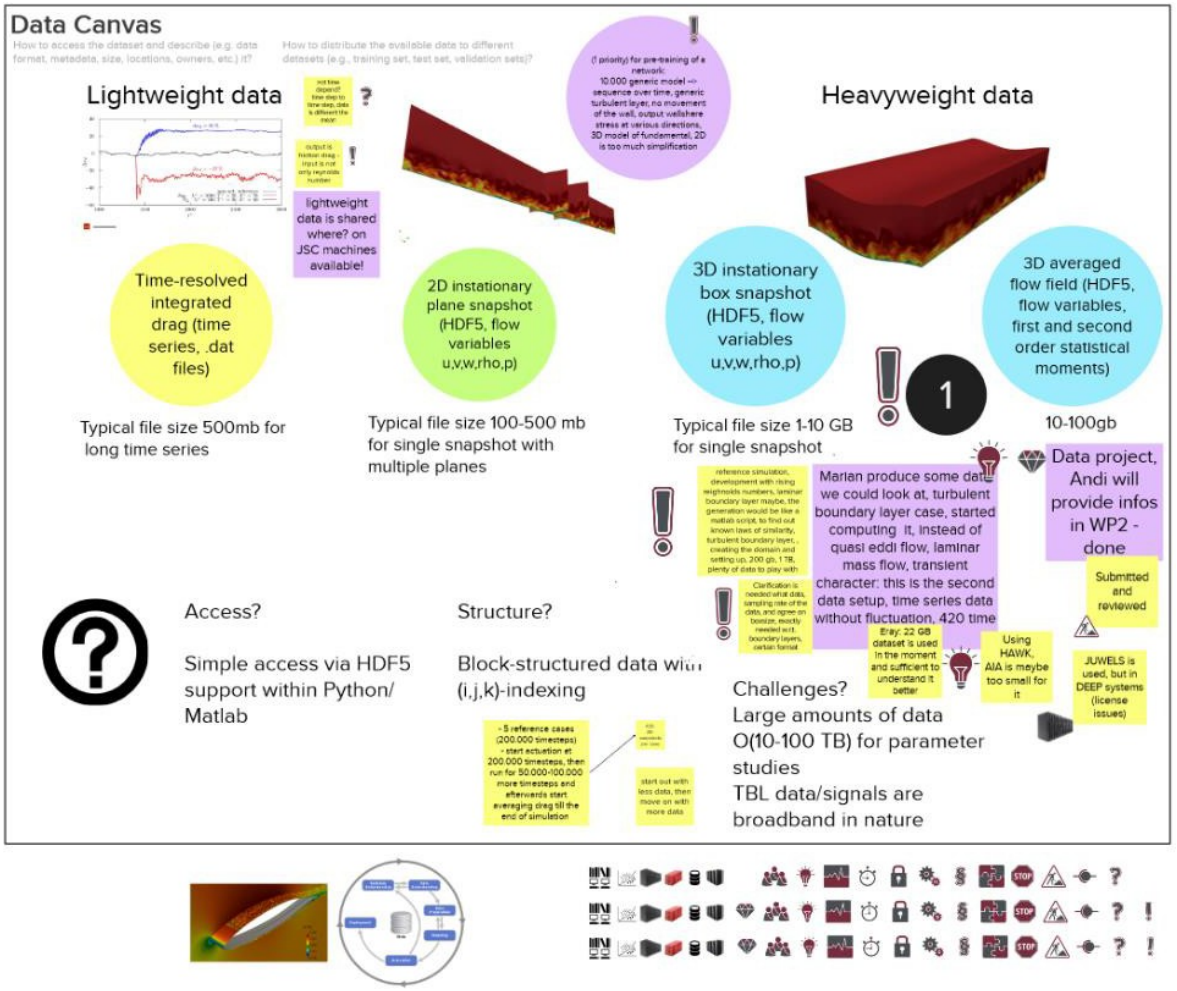

Figure 4: Mural board example of WP3 Task 3.1 – Data Canvas.

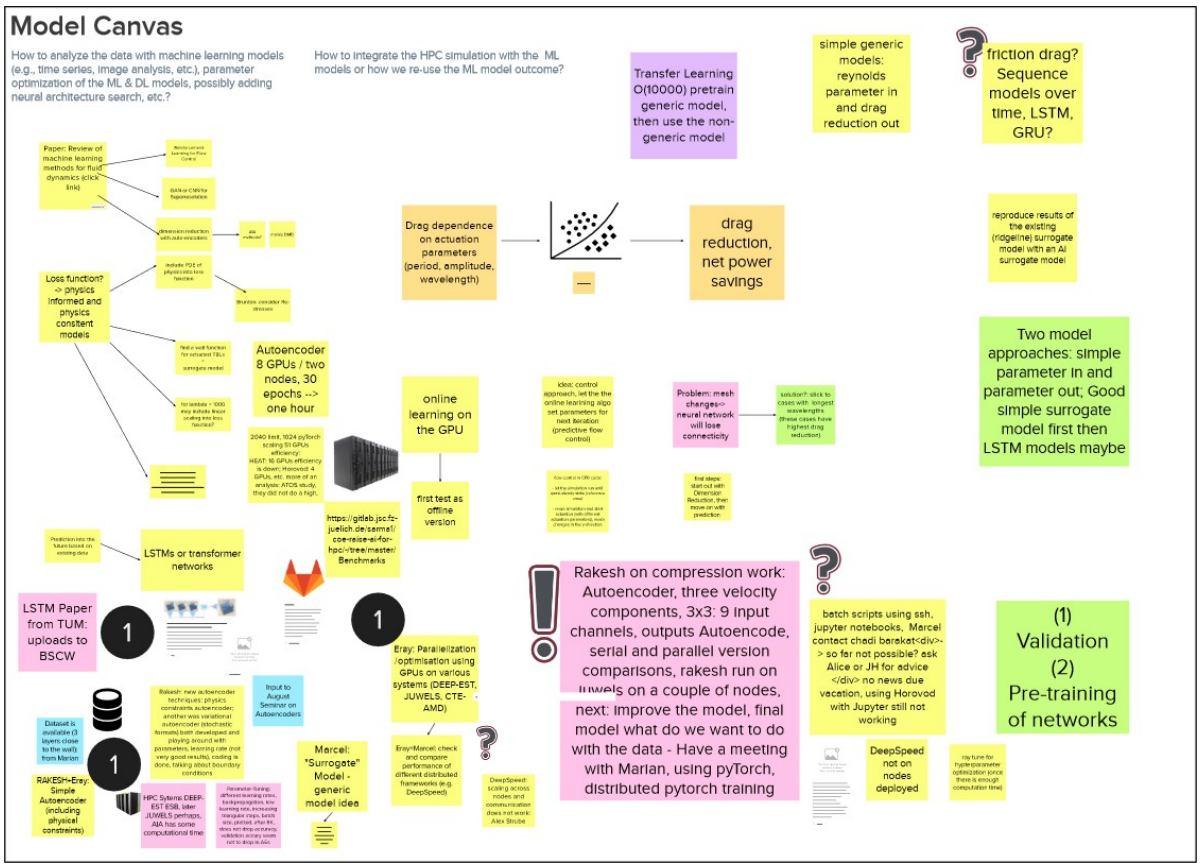

Figure 5: Mural board example of WP3 Task 3.1 – Model Canvas

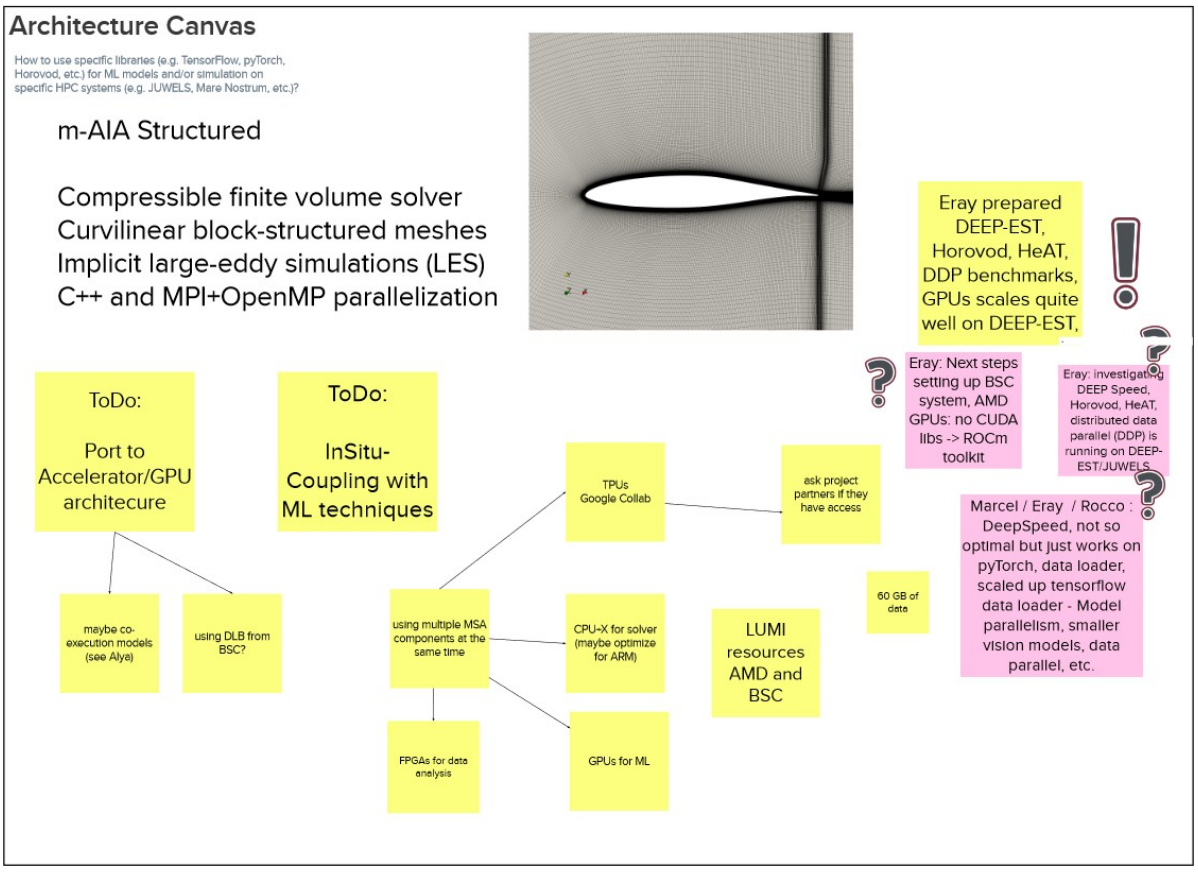

Figure 6: Mural board example of WP3 Task 3.1 – Architecture Canvas

#	Canvas	Purpose
1	Problem	Identify what the research questions, boundary conditions, abstractions, assumptions, quality requirements, AI usage, etc. are.
2	Data	Identify how to access the dataset and describe it (e.g., data format, metadata, size, locations, owners).
3	Model	Identify how to analyze the data with ML models (e.g., time series, image analysis), parameter optimization of the ML & DL models, possibly adding neural architecture search, etc.
4	Architecture	Identify how to use specific libraries (e.g., TensorFlow, pyTorch, Horovod) for ML models and/or simulation on specific HPC systems (e.g., JUWELS, Mare Nostrum).

Table 1: Description of the four different Canvas areas of Mural boards

3.2 Identified requirements

The requirements for a unique AI framework in RAISE are driven by four different factors (see Section 2): (1) *hardware infrastructure*, (2) *software infrastructure*, (3) *WP3 use cases*, and (4) *WP4 use cases*. As a whole, they contribute to use-case-specific HPC & AI technical requirements with respect to software and hardware of a unique AI framework, ready for Exascale.

3.2.1 Identified AI / HPC methods as AI model requirements from use cases

One of the most important goals of CoE RAISE is to support AI towards Exascale using cutting-edge HPC systems and as such building, training, and evaluating ML and DL models is considered most important.

The factsheet development process and the design and discussion sessions in the continuously accessible virtual Interaction Rooms enabled the identification of an initial set of relevant AI/HPC methods that have been documented in the RAISE milestone MS2 – AI/HPC Methods (achieved in project Month 7). A summary of these methods and the MS2 is shown in Table 2. A key requirement of the RAISE AI framework is thus to support the design, development, and use of AI models of this table. In other words, Table 2 represents the requirements analysis with respect to the model side of the AI framework. Its DL libraries, packages, and tools need to support these models.

One of the key challenges in AI modeling that can leverage HPC systems, particularly on the Exascale level, is the tuning of hyper-parameters of AI models (e.g., choice of optimizers, number of layers in DL networks, resolutions of input sizes, number of neurons in layers). Hence, orthogonal requirements to these model-specific requirements summarized in Table 2 are AutoML techniques and NAS methods that enable CoE RAISE use case developers - with a helper tool - to find the right set of hyper-parameters for their model.

Use Case	**AE**	**PIML**	**ANNs**	**CNN**		**NO**	**SMs**			**GNN**	**IN**	**LSTM**	**GRU**
Details	***CAE***		***RBF-ANN***	***U-Net***	***RESNET***	***FNO***	***AR***	***ARMA***	***ARIMA***		***JEDI-net***		
AI for turbulent boundary layers	X	X											
AI for wind farm layout optimization			X				X	X	X				
AI for data-driven models in reacting flows				X						X			
Smart models for next generation aircraft engine design				X						X			
AI for wetting hydrodynamics						X							
Event reconstruction and classification at the CERN HL-LHC use case										X	X		
Seismic imaging with remote sensing for energy applications	X				X								
Detect-free metal additive manufacturing	X				X								
Sound Engineering												X	X

Table 2: Compact AI and HPC Methods overview as use case requirements (the abbreviations can be found at the end of this document).

3.2.2 AI software and HPC hardware infrastructure requirements

Apart from the above listed AI/HPC methods and model requirements per use case, there exist further software and hardware infrastructure requirements that have been identified during the Interaction Room process. As they have been identified across many use cases, the use cases that raised them are not listed in the following Table 3. They are rather seen as general design requirements for the RAISE AI framework. The requirements are numbered as Requirement #X (RQX) etc. to reference them in Section 4 of this document, where the overall AI framework design is described, and to explain how RAISE aims to address those requirements.

#	Description	Level	Affected Technologies
RQ1	Applications need to run on different HPC systems without detailed knowledge of underlying package versions.	Hardware Software	DL libraries, e.g., TensorFlow, Keras, PyTorch, Horovod.
RQ2	Proven scalability of the framework components.	Software	Horovod and PyTorch-DDP should have shown good scaling capabilities, see findings in Section 3.3.
RQ3	Support for ONNX for TL and model sharing.	Software	DL models need to store and re-use ONNX models.
RQ4	High-level access via Jupyter notebook for DL modeling including Kernels aware/choice of HPC modules.	Software	Jupyter notebook and JupyterLab environments should be supported by the framework.
RQ5	Low-level access via batch submission enabling automation and scaling of multi-GPU setups.	Software	SSH protocols to enable low-level access to HPC systems using their batch schedulers.
RQ6	Enable reproducibility by using open-source description of data science tasks and AI models.	Software	MLFlow and ClearML and associated deployments on HPC systems re-using container technologies.
RQ7	Enable support for container technologie to enable portability between different HPC centers with different AI stacks.	Software	Technologies like Singularity and Docker enable the portability between HPC systems and are used in HPC centers, e.g., at JSC and BSC-CNS.
RQ8	The framework should be agnostic with respect to accelerator types to use it with	Hardware	Accelerators, e.g., NVIDIA GPUs, are instrumental for DL and towards Exascale it is expected to

	DL applications without knowing the details of different accelerator types and underlying libraries.		leverage other accelerator types as well, e.g., AMD Instinct MI100.
RQ9	The framework needs to support cutting-edge I/O capabilities for high performance and be able to work with large datasets.	Hardware	Towards Exascale, large quantities of datasets are expected and the I/O capabilities of the framework need to leverage the underlying hardware infrastrastructure.

Table 3: General requirements for an AI framework for Exascale

3.3 Selected benchmark activities

The tasks mostly contributing to this Deliverable (Task 2.4: Software design of a unique AI framework and Task 2.5: Cross-Sectional AI Methods) are complemented by other relevant tasks in WP2 (Task 2.1: Modular and heterogeneous supercomputing architectures, Task 2.2: Hardware prototypes, and Task 2.3: Benchmarking on disruptive technologies) that support the technology review with respect to scaling and thus influence the implementation of the AI framework (e.g., RQ2 in Table 3). This includes benchmark activities of DL frameworks as shown in the figures below whereby results will be reported in other WP2 Deliverables (e.g., D2.2: Report on porting and performance engineering activities at Month 12). As shown in Figure 7, this includes measurements of data throughput from relevant frameworks like PyTorch with Horovod compared to PyTorch-DDP. RAISE started to publish results with a submitted paper (not accepted yet) [7]. The figures below show the trade-off of using PyTorch with Horovod for accuracy or PyTorch-DDP for efficiency as shown in Figure 8. Using an increase in batchsize (BS) that is required by increasing the number of GPUs for scaling, a quick drop in validation accuracy in PyTorch-DDP is observed while Horovod remains better until 16K (see Figure 9).

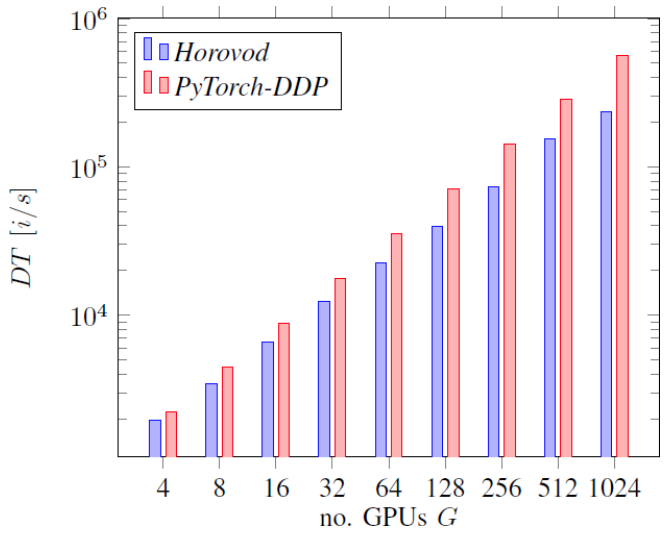

Figure 7: Comparison of the data throughput (DT) in images i per second [7].

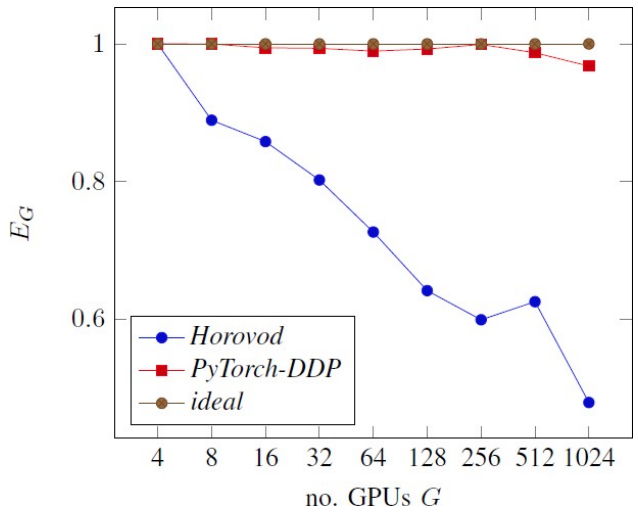

Figure 8: Comparison of the parallel efficiency E [7].

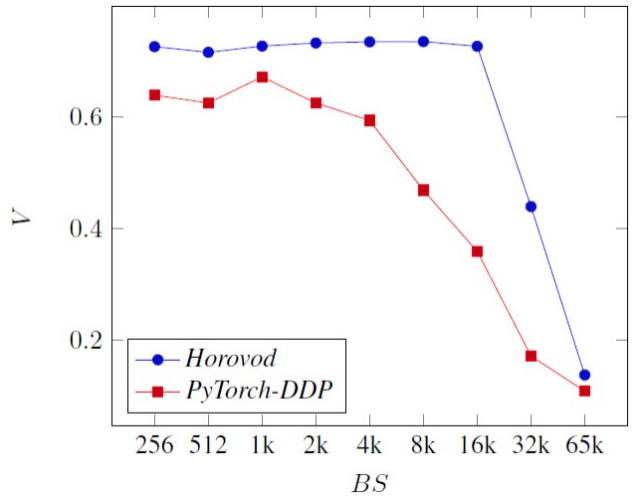

Figure 9: Accuracy on the validation set V over the overall BS [7].

4 AI Framework Software Layout Plan

This section focuses on the software design of a unique AI framework that is co-designed by the different application use cases of WP3 and WP4 with respect to AI model requirements and overall design and usability requirements elaborated in Section 3. Based on the lessons learned from the Interaction Rooms and the definition of relevant requirements in Section 3, this section reveals an architectural blueprint of the proposed RAISE reference model and its associated architectural elements. The subsequent sections provide several pieces of architecture work (e.g., reference architecture, technologies for implementation, and de-facto standards) that are all specifically designed to support AI applications towards Exascale.

4.1 Generalized AI framework layout as Reference Model design

After an extensive requirements analysis based on factsheets and using the Interaction Room in regular meetings, WP2 has decided together with WP3/WP4 on an initial layout plan for the AI framework as a reference model to be used in RAISE. This plan addresses the initially identified requirements of the WP3 and WP4 use cases. The reference model is used as an 'umbrella term' for the WP2 activities, including its associated architecture elements. The definition of the abstract reference model itself is the focus of this section as shown in Figure 10 (marked in red). Associated architectural work elements are defined in subsequent sections. Another principle applied here is that the reference model defines 'entities' for various functionality and 'relationships' between them. The overall concept of the abstract design takes the requirements of Section 3 into account. It is 'abstract', because its entities and concepts are an abstract representation of the entities that often exist in AI & HPC use cases in RAISE. One example is that the framework should abstract from specific ML and DL frameworks like TensorFlow and PyTorch, and thus an abstract representation is used in the architecture description following software engineering principles of reference models. The specific realizations and reference architectures below then have the individual technologies as blueprints for specific use case architectures that use the AI framework.

To better understand RAISE's unique AI framework, it is helpful to provide a general abstraction in the sense of 'entities' that play a key role in the realization of the software framework or its usage by RAISE use cases. They enable a general blueprint and design foundation of a unique AI framework in RAISE. These entities refer to

- *HPC systems,*
- *module environments with libraries in specific versions,*
- *different user roles with security credentials,*
- *dedicated AI scripts or intertwined AI code with simulation sciences code,*
- specific *user access methods,* and
- *external community exchange tools.*

The *external community exchange tools* refer to components defined in the RAISE framework that are externally hosted (e.g., for data sharing with the larger AI & HPC community). The entities are highly dependent on each other, meaning that *HPC system* entities only offer specific *module environments with libraries in specific versions*. As such, their n:n relationship is complex when considering the rich amount of *dedicated AI scripts or intertwined AI code with simulation sciences code*.

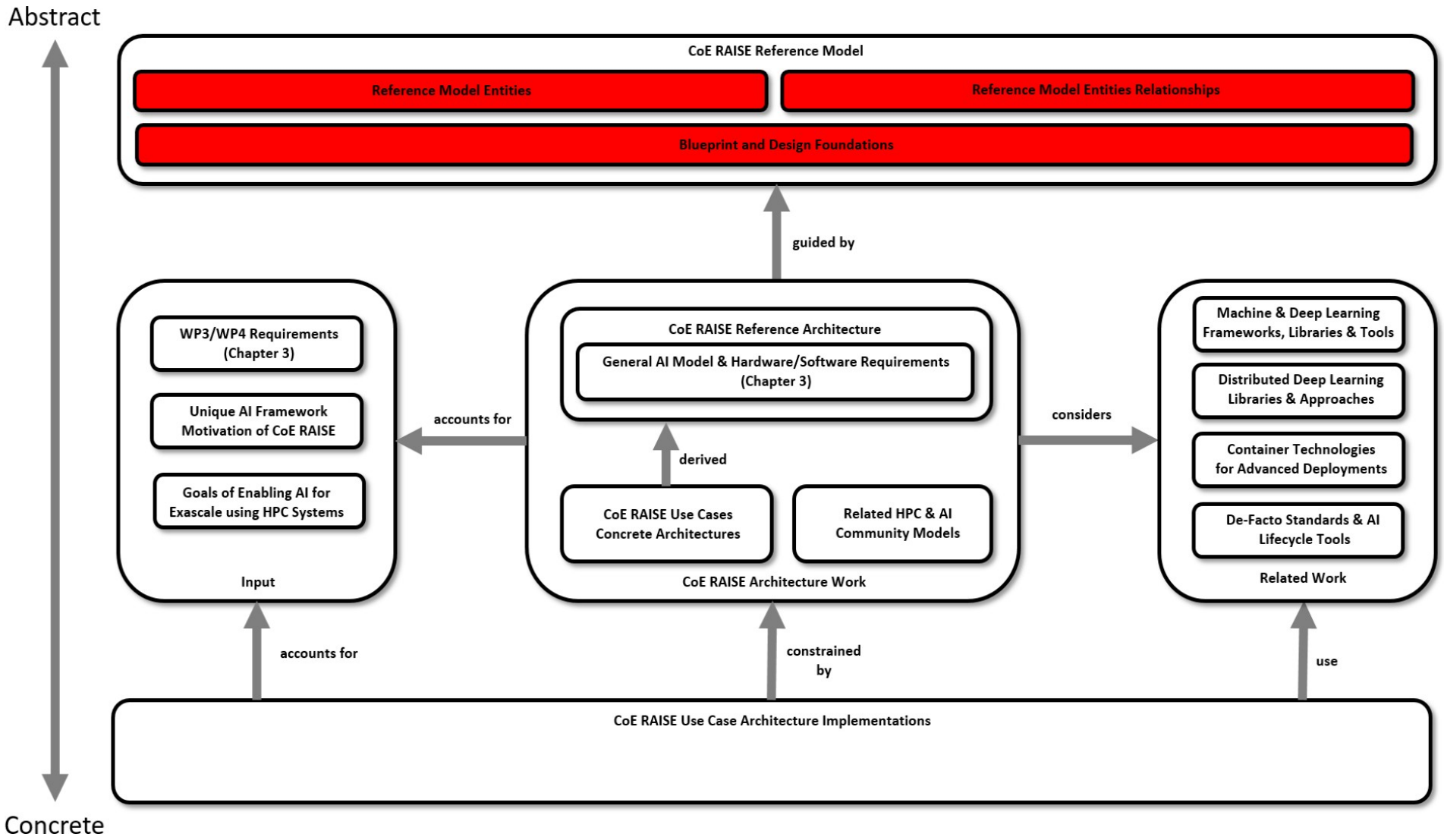

Figure 10: The AI framework abstract reference model.

In addition, the relationship of *user access methods* can vary from low-level access (e.g., SSH account) to high-level access (e.g., Jupyter notebooks), also depending on the *different user roles with security credentials*. All these general elements of the RAISE reference model guide the blueprint and design foundations of the following RAISE reference architecture.

4.2 Associated Reference Architecture general design

The previous section provides the RAISE reference model design. The design is abstract and thus not detailed enough for implementation. It only allows for a general understanding with respect to software and hardware ecosystems involved and a comparison between other reference models emerging in the large AI & HPC community. As shown in Figure 10, its associated RAISE reference architecture therefore provides much more details for the implementation of this model, showing inputs (red) of RAISE and identifying relevant related work (red) that are listed in Section 2 as relevant technologies.

Figure 11 shows the RAISE reference architecture with ML/DL frameworks, distributed DL libraries, container technologies, and de-facto standards & AI lifecycle tools as core building blocks.

Guided by the abstract RAISE reference model design, its entities are mapped to specific core building blocks of the RAISE reference architecture, as shown in Figure 12. This initial blueprint of the RAISE reference architecture represents the generalized layout of the AI framework of RAISE with approved core building blocks for Exascale applications in the future. That informal approval is performed internally in RAISE by the benchmark teams and from other tasks in WP2 that in parallel perform technology assessments. Only if tools are able to scale well, they will be supported in the framework.

As a software framework, still taking into account hardware infrastructure constraints, it provides a ‘standard way’ to build and deploy AI applications in RAISE and is a universal, reusable software environment that provides particular functionality as part of a larger software platform, e.g., including specific HPC system module deployments, to facilitate the development of AI use cases in RAISE. Based on the requirements, the software framework includes supporting programs, code libraries, toolsets, and APIs that bring together all the different components to enable development of AI models of *WP3 Compute-Intensive CoE RAISE Use Cases* (Figure 12, item A) and *WP4 Data-Intensive CoE RAISE Use Cases* (Figure 12, item B). One of the goals of the CoE RAISE Reference Architecture is also to enable other *Exascale HPC & AI Community Use Cases* (Figure 12, item C) such as those driven by other CoEs or future Digital Twins such as Destination Earth[31].

To address the requirements stated in Section 3, the RAISE reference architecture needs to support low-level (RQ5) and high-level access methods (RQ4). In this initial blueprint, the core building block to enable low-level access is via *SSH protocols using batch scheduler scripts for automation* (Figure 12, item D). The requirement analysis of the WP3 and WP4 use cases revealed that an interactive access via *Jupyter notebooks* is also required (Figure 12, item E) to enable quick and rapid prototyping of DL algorithms. In this context, it is possible to create SSH sessions out of Jupyter notebooks on HPC systems.

[31] Destination Earth: https://digital-strategy.ec.europa.eu/en/policies/destination-earth

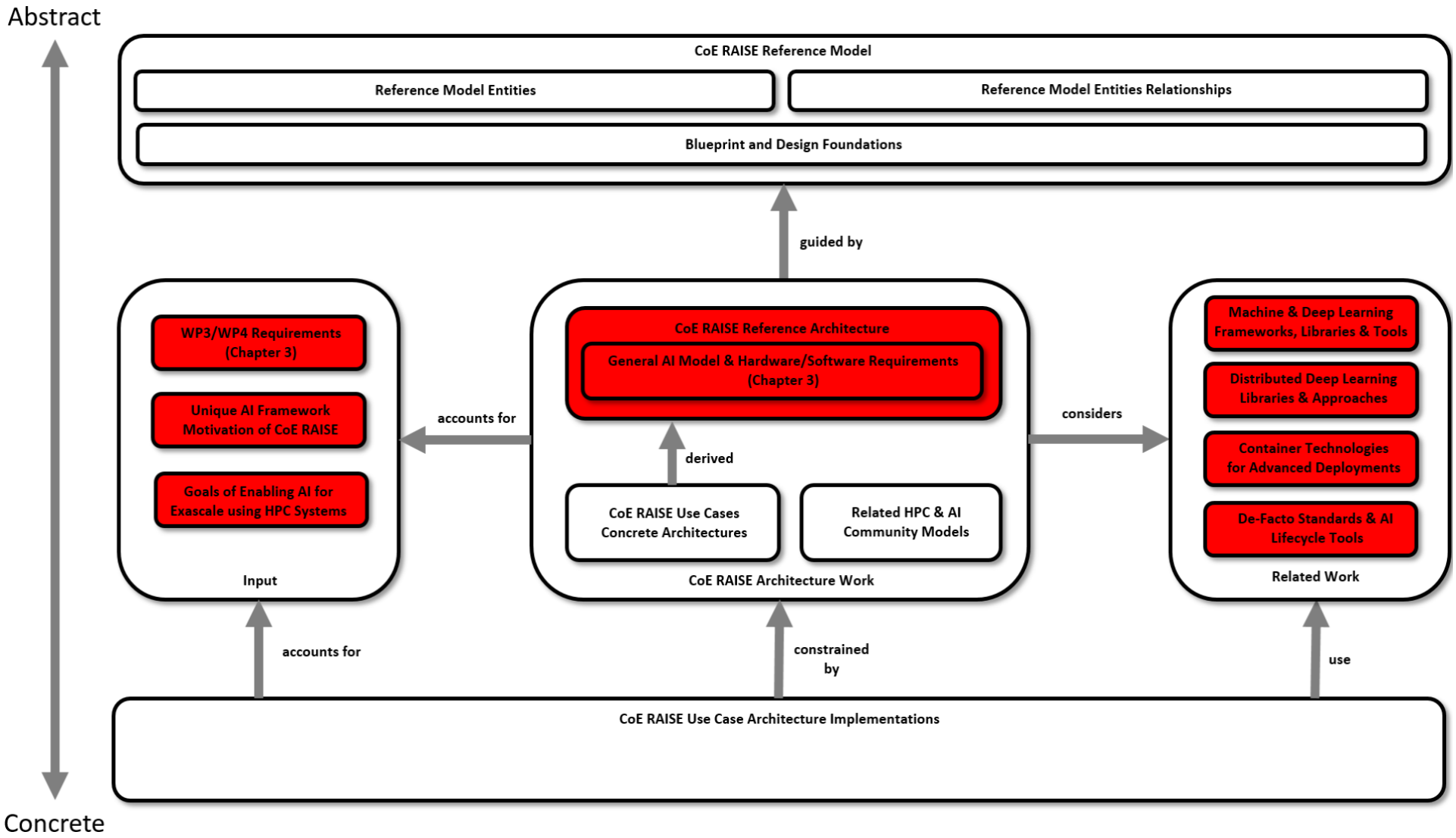

Figure 11: The CoE RAISE reference architecture.

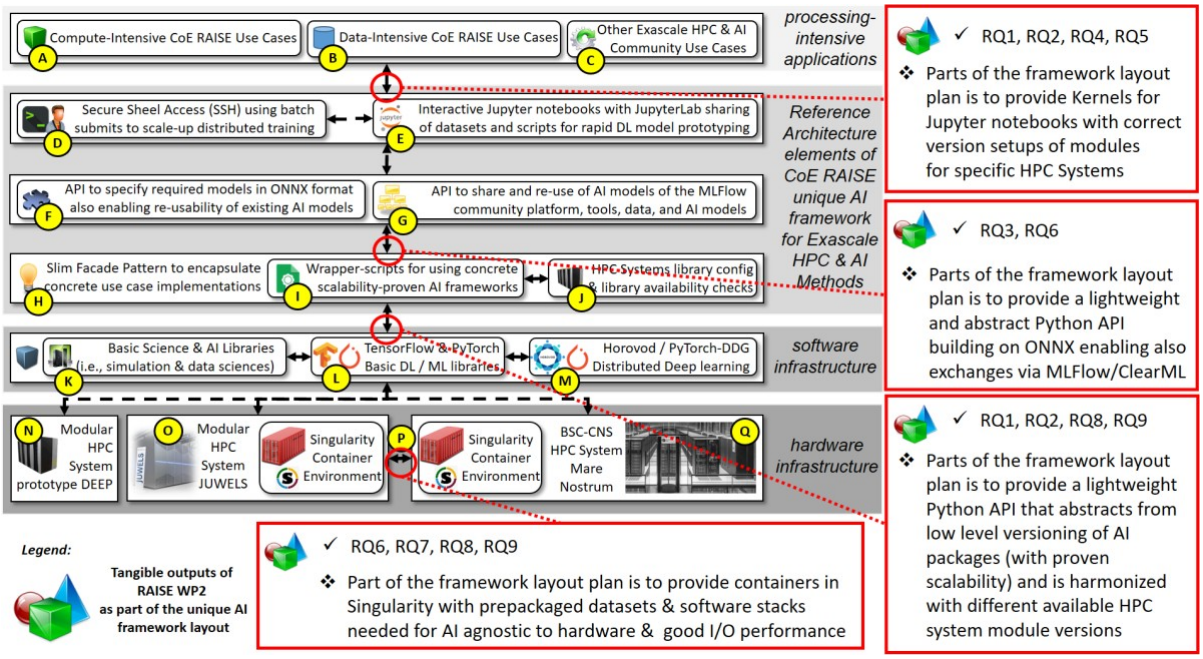

Figure 12: The CoE RAISE reference architecture and its core building blocks

The framework needs to be hardware-agnostic with respect to accelerators (RQ8) and requires a high I/O performance capable of working with large quantities of data (RQ9). To enable fast portability between different DL frameworks and reproducibility (RQ6), the unique AI framework needs to abstract the specification of specific tasks (RQ2) by using the *ONNX format* (Figure 12, item F). A more comprehensive view on use cases reveals the requirements that platforms like *MLFlow* (Figure 12, item G) should be supported to share and re-use existing AI models among the broader AI & HPC community (RQ6).

To map the above rather abstract specifications to specific software and hardware infrastructure via the *Facade pattern* (Figure 12, item H), the unique AI framework layout design employs an abstract wrapper functionality that *maps the abstract specifications to concrete software and hardware configurations* (RQ1) of available HPC systems (Figure 12, item I), encapsulating users from the need to know about low-level version details. In this context, it is important to check what versions are available in what modules on the specific *HPC systems* (Figure 12, item J) as RAISE's use cases considered this a major obstacle for using AI technologies on HPC systems today.

The reference architecture in Figure 12 includes specific versions of packages within its software infrastructure provided by HPC centers. Those software packages are *basic science libraries* (Figure 12, item K) as mentioned in the background, e.g., NumPy, Pandas. The RAISE reference architecture will also leverage specific harmonized versions of the DL packages *TensorFlow and PyTorch* (Figure 12, item L), as well as *PyTorch-DDP and Horovod for scaling towards Exascale* (Figure 12, item M). At the time of writing, the complementary benchmark activities of WP2 investigate scaling and its limits when using these packages particularly with respect to their scalability towards Exascale.

The reference architecture in Figure 12 also includes a hardware infrastructure that deploys the software infrastructure above and that is accessible to RAISE use case members. Among the HPC systems available to RAISE are the *DEEP prototype* (Figure 12, item N), the *JUWELS system at JSC* (Figure 12, item O), the *MareNostrum system at BSC-CNS* (Figure 12, item Q),

and others. To enable portability, users of the AI framework for RAISE based on this reference architecture are able to use specially prepared containers with use-case specific software stacks prepared based on *Singularity* (Figure 12, item P).

The tangible output of WP2 as contributions to the unique AI framework for Exascale based on the described reference architecture are as follows:

- provide kernels for Jupyter notebooks with correct version setups of modules for specific HPC systems addressing RQ1, RQ2, RQ4, and RQ5
- provide a lightweight and abstract Python API building on ONNX to enable easy exchange with MLFlow/ClearML addressing RQ3 and RQ6
- provide a lightweight Python API that abstracts from low level versioning of AI packages (with proven scalability) and is harmonized with different available HPC system module versions addressing RQ1, RQ2, RQ8, and RQ9
- provide containers in Singularity with prepackaged datasets and required software stacks needed for AI models addressing RQ6 and RQ7. This environment needs to be able to work with large-scale datasets and having good I/O performance addressing RQ9. Those environments need to be underlying hardware-agnostic addressing RQ8

The key goal is thus not to re-invent the wheel of existing frameworks but rather focus on the problem areas, e.g., to correct kernels for HPC systems in Jupyter, and provide an overall better usability experience for HPC exascale users. It is thus going beyond the pure ML/DL algorithms and aims to offer the use cases an overall improved AI experience on HPC platforms and make the use of them more seamless.

To sum up, the CoE RAISE Reference Architecture includes all necessary components and core building blocks required by the RAISE use cases, and also enables the adoption of the larger uptake of the HPC & AI community with similar use cases. The initial generalized framework is illustrated in Figure 12 while updates are foreseen in the the project runtime in subsequent Deliverables such as D2.13 in Month 26, which will provide an update to this document, or related Deliverables such as the D2.14 report on novel use-case-overarching AI technologies in Month 12. Given that many technology evaluations and benchmarks are still underway in a very lively research field of popular DL tools, several core building blocks might change or be extended to future emerging HPC & AI ecosystem impacts, e.g., disruptive technologies such as quantum computing).

4.3 Specific architectures and implementation of use cases

The necessary next step as part of this Deliverable upgrade in the next reporting period (project Month 28) is to provide evidence of how all the previous abstract and theoretical considerations lead to practical impact on RAISE's use cases. The impact is shown in terms of technology improvements as well as in terms of scientific innovation through real CoE use cases that take advantage of the unique AI framework and that demonstrate Exascale potential. This moves the focus one step further from the abstract to the more concrete.

But the previous rather theoretical work ensures that the developments in the use cases are in line with a future integration into a generalized software framework by continuously monitoring them in other WP2 Deliverables. e.g., in the D2.10 report on monitoring processes towards a unique AI framework in Month 18. A lively interaction using the Interaction Room with AI framework designers and the use case developers was necessary to find a general layout, which is suitable for a generalized approach. Interfaces and APIs are described in detail

together in subsequent Deliverables with the use case providers to ensure the generalized approach and future sustainability of the framework as well as its easy application is achievable in RAISE use cases.

The specific architecture and implementations go beyond the scope of this initial software layout Deliverable D2.12 at project Month 9. In addition, WP2 factsheets will continuously be updated for capturing the RAISE use case architecture implementations (marked in red in Figure 13) based on specific architectures (red) and potential impacts on related HPC & AI community models (red). D2.12, however, will be continuously updated with a major update in project Month 28 (D2.13). The continued work on the unique AI framework together with WP3 and WP4 use cases will be driven forward with a living architecture document during the project runtime. That WP2 architecture work will be a guide used in subsequent Interaction Room sessions with members of WP3 and WP4 to create derived specific use case architectures.

Figure 13 shows how tangible CoE use case architctures of the CoE reference model are derived from the abstract CoE reference architecture taking advantage of the CoE RAISE unique AI framework design. The overall approach can be compared to the abstract International Organization for Standardization / Open Systems Interconnection (ISO/OSI) Reference Model and its specific reference architecture via Transmission Control Protocol (TCP)/Internet Protocol (IP).

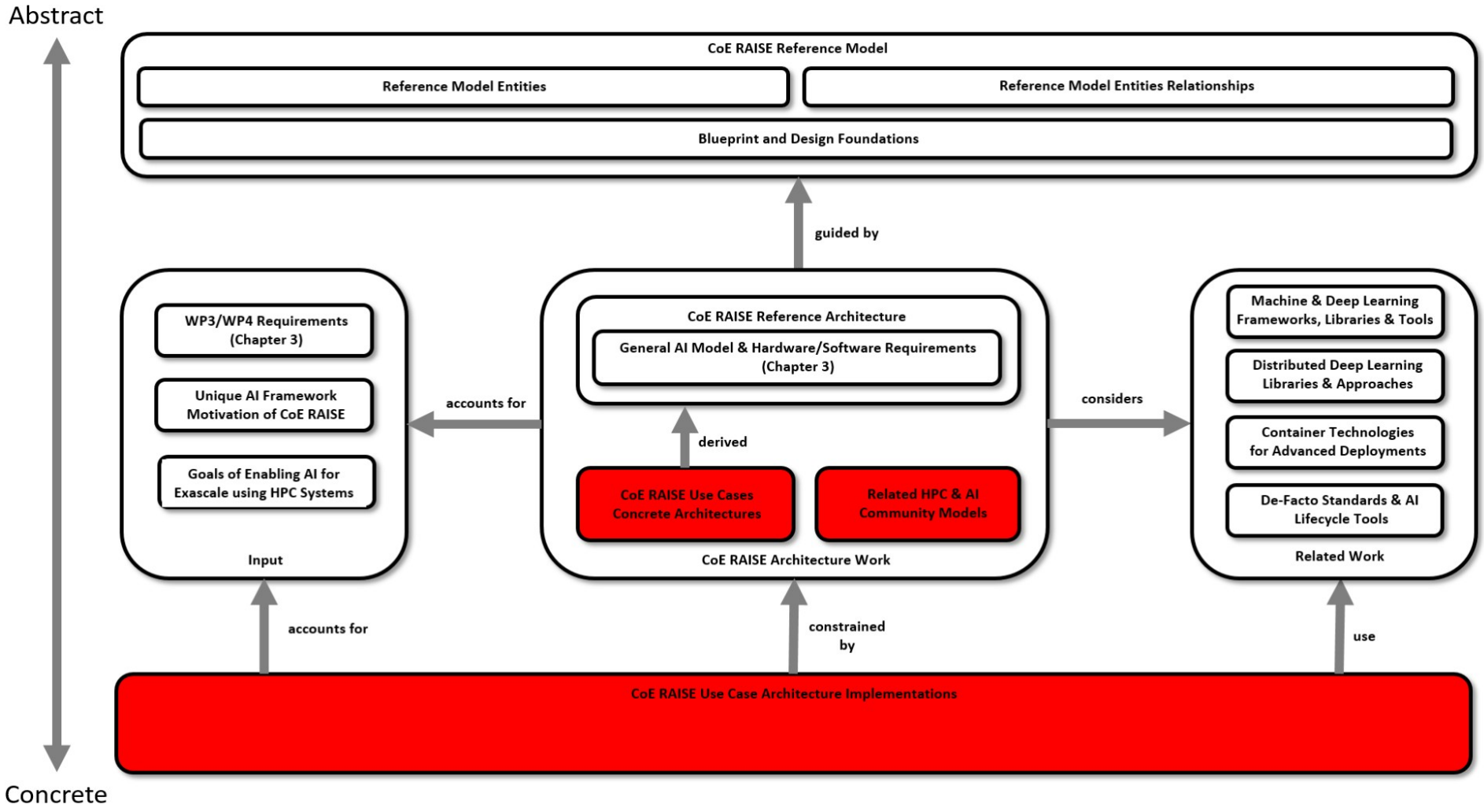

Figure 13: Specifiic use case architectures derived from the Reference Architecture.

5 References

[1] CoE RAISE Deliverable D2.1 Best practice guidelines/tutorials for MSA/heterogenous systems

[2] CoE RAISE Deliverable D2.5 Best practice guidelines/tutorials prototype

[3] Ben-Nun, T. and Hoefler, T., 2019. Demystifying parallel and distributed deep learning: An in-depth concurrency analysis. *ACM Computing Surveys (CSUR)*, *52*(4), pp.1-43. https://dl.acm.org/doi/pdf/10.1145/3320060

[4] A. Sergeev and M. D. Balso, “Horovod: fast and easy distributed deep learning in tensorflow,” 2018, arXiv:1802.05799. https://arxiv.org/abs/1802.05799

[5] S. Li, Y. Zhao, R. Varma, O. Salpekar, P. Noordhuis, T. Li, A. Paszke, J. Smith, B. Vaughan, P. Damania, and S. Chintala, “PyTorch distributed: experiences on accelerating data parallel training,” Proceedings of the VLDB Endowment, vol. 13, no. 12, pp. 3005–3018, 2020. https://dl.acm.org/doi/pdf/10.14778/3415478.3415530

[6] Riedel, M., Sedona, R., Barakat, C., Einarsson, P., Hassanian, R., Cavallaro, G., Book, M., Neukirchen, H. and Lintermann, A., 2021, June. Practice and Experience in using Parallel and Scalable Machine Learning with Heterogenous Modular Supercomputing Architectures. In *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (pp. 76-85). IEEE. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9460702

[7] Aach, M, Inanc, E., Sarma, R., Riedel, M., and Lintermann, A., ‘Performance Analysis of Horovod and PyTorch-DDP on the JUWELS Supercomputer‘, submitted to the IEEE Big Data Conference 2021

List of Acronyms and Abbreviations

AE	Auto-Encoder
AD	Actuator Disc
AI	Artificial Intelligence
ANN	Artificial Neural Network
API	Application Programming Interface
AR	Auto-Regressive
ARMA	AR Moving Average
ARIMA	AR Integrated Moving Average
BS	Batch Size
BSC	Barcelona Supercomputing Centre, Spain
BSCW	Basic Support for Cooperative Work
CAE	Convolutional Auto-Encoder
CERN	European Organisation for Nuclear Research / Organisation Européenne pour la Recherche Nucléaire, International organization, Switzerland
CNN	Convolutional Neural Networks
CoE	Center of Excellence
CoE RAISE	Center of Excellence "Research on AI- and Simulation-Based Engineering at Exascale"
DA	Data Augmentation
DALI	NVIDIA Data Loading Library
DDP	see PyTorch-DDP
DEEP	Dynamical Exascale Entry Platform (project FP7-ICT-287530)
DL	Deep Learning
DNN:	Deep neural network
DT	Data Throughput
Exascale:	Computer systems or Applications, which are able to run with a performance above 10^{18} Floating point operations per second
FM	Flanders Make VZW
FNO	Fourier Neural Operator
FZJ	Forschungszentrum Jülich GmbH, Jülich, Germany
GNN	Graph Neural Networks
GPGPU	General Purpose Graphics Processing Unit
GPU	Graphics Processing Unit
GRU	Gated Recurrent Units
HeAT	Helmholtz Analytics Toolkit
HL	High Luminosity
HPC	High-Performance Computing
HPDA	High-Performance Data Analytics
HTC	High-Throughput Computing
IN	Inception Network
IP	Internet Protocol
ISO/OSI:	International Organization for Standardization / Open Systems Interconnection
JSC	Jülich Supercomputing Centre
LHC	Large Hadron Collider, the world's most powerful accelerator providing research facilities for High Energy Physics researchers across the globe
LSTM	Long Short-Term Memory

ML	Machine Learning
MS	Milestone, followed by a number, term to designate a milestone number in the CoE RAISE project
MSA	Modular Supercomputer Architecture
NAS	Neural Architecture Search
NLTK	Natural Language Toolkit
NO	Neural Operator
ONNX	Open Neural Network Exchange
ParTec	ParTec AG, Munich, Germany. Linked third Party of FZJ in CoE RAISE
PIDL	Physics-Informed DL
PIML	Physics-Informed ML
PyTorch-DDP	PyTorch Distributed Data Parallel
RESNET	Residual Neural Network
RBF	Radial Basis Function
RNN	Recurrent Neural Network
RL	Reinforcement Learning
RQ	Requirement
RS	Remote sensing
RTU	Rigas Tehniska Universitate, Latvia
SM	Statistical Models
TCP	Transmission Control Protocol
TL	Transfer learning
UOI	Háskóli Íslands – University of Iceland, Iceland
WP	Work Package