D4.13 – i4Q AI Models Distribution to the Edge

WP4 – BUILD: Manufacturing Data Analytics for Manufacturing Quality Assurance

i4Q has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958205.

Document Information

GRANT AGREEMENT NUMBER	958205	ACRONYM	i4Q	
FULL TITLE	Industrial Data Services for Quality Control in Smart Manufacturing			
START DATE	01-01-2021	DURATION	36 months	
PROJECT URL	https://www.i4q-project.eu/			
DELIVERABLE	D4.13 – i4Q AI Models Distribution to the Edge V2			
WORK PACKAGE	WP4 – BUILD: Manufacturing Data Analytics for Manufacturing Quality Assurance			
DATE OF DELIVERY	CONTRACTUAL	31-Dec-2022	ACTUAL	30-Dec-2022
NATURE	Other	DISSEMINATION LEVEL	Public	
LEAD BENEFICIARY	IBM			
RESPONSIBLE AUTHOR	Benny Mandler (IBM)			
CONTRIBUTIONS FROM				
TARGET AUDIENCE	1) i4Q Project partners; 2) industrial community; 3) other H2020 funded projects; 4) scientific community			
DELIVERABLE CONTEXT/ DEPENDENCIES	This document presents a technical overview of the i4Q AI Models Distribution to the Edge (i4Q^{AI}) component. This document has a preceding document (D4.5) and is a second and final iteration of this deliverable and task, scheduled for M24 (Dec. 2022).			
EXTERNAL ANNEXES/ SUPPORTING DOCUMENTS	None			
READING NOTES	None			
ABSTRACT	This task addresses a multi-tier infrastructure to cope with the management of AI-based models in a hybrid cloud edge manufacturing environment. One of the major challenges in this area is scalability. The goal of this task is to reach a high scale by using techniques such as a policy-based distribution mechanisms. This task supports AI at the edge operations, enabling developers to concentrate on their domain of expertise while this solution provides an automatic placement and deployment apparatus. The AI models deployment task is highly coordinated with the workload distribution mechanism such that AI workloads make use of the corresponding AI models.			

Document History

VERSION	ISSUE DATE	STAGE	DESCRIPTION	CONTRIBUTOR
0.1	07-Nov-2022	ToC	Table of Contents creation	IBM
0.2	01-Dec-2022	Draft	First draft available for internal review	IBM
0.3	12-Dec-2022	Internal Review	Internal review	WHI, AIMPLAS
0.4	19-Dec-2022	Draft	Incorporate comments from external review	IBM
1.0	30-Dec-2022	Final Document	Final quality check and issue of final document	CERTH

Disclaimer

Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Copyright message

© i4Q Consortium, 2022
This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS/ACRONYMS

ACM	Advanced Cluster Management
ACM	Advanced Cluster Management
AI	Artificial Intelligence
CNC	Computerized Numerical Control.
DSS	Decision Support System
EW	Edge Workload
IoT	Internet of Things
K3s	Lightweight Kubernetes
K8s	Kubernetes
LRT	Line Reconfiguration Tool
NFR	Non-Functional Requirement
ML	Machine Learning
OCM	Open Cluster Management
RHACM	Red Hat Advanced Cluster Management
UI	User Interface
VM	Virtual Machine

Executive summary

Deliverable D4.13 presents a technical overview of the i4Q AI Models Distribution to the Edge solution. This deliverable provides an overview of the unique operational environment of AI on the edge, diving into the specific capabilities provided by this solution to other i4Q solutions and pilots.

This task addresses a multi-tier infrastructure designed for the management of AI-based models in a hybrid cloud-edge manufacturing environment. Scale is one of the main areas of concern in such environments, thus operations are planned to be automated as much as possible by employing techniques like policies and labels-based distribution and deployment. There is a tight coordination required between this task and the AI workload distribution mechanism (implemented in the solution i4Q^{EW}: AI Workload Placement and Deployment), such that the AI workload and model shall meet and collaborate in the correct edge targets nodes.

This solution resides at the back-end infrastructure level providing capabilities to be used by other solutions and pilots. The design is influenced by internal functionality at the core of the solution as well as interfaces and capabilities required by potential clients. This solution shall provide support for different kinds of model wrappers to be deployed, for instance native binary to be loaded by the corresponding workload or a wrapping http server whose interfaces can be invoked by the corresponding AI workload. In addition, the model to be deployed can be prepared in several flavors such as a standalone container or wrapped within a helm chart.

For the first release of this solution the main collaboration would be with the manufacturing Line reconfiguration (LRT) solution provided by UPV in the context of the FACTOR pilot. This solution shall deploy the models provided by the LRT solution. As a second stage, we aim for the solution to be incorporated within the generic pilot as well.

A recorded video demonstrating the current version is being made available on the project repository in SharePoint and may be available for dissemination purposes.

This document i4Q D4.13 v2 is an update of v1 of D4.5., for this reason it contains information of the 1st version together with the updates developed in this 2nd version.

Document structure

Section 1: is comprised of a general description of the operational environment (cloud-edge multi-tier environment and challenges) and specifically the **i4Q AI Models Distribution to the Edge** component, providing an overview and a list of features provided by this solution. It is addressed to additional solutions and end-users who may be interested in making use of this i4Q Solution from within another solution or a pilot.

Section 2: Contains the technical specifications of the **i4Q AI Models Distribution to the Edge** component, providing high level view and its accompanying architecture diagram (from the reference architecture). It is addressed to software developers who may want to interact with this solution.

Section 3: Details the implementation status of the **i4Q AI Models Distribution to the Edge** component, explaining the current status.

Section 4: Provides the conclusions.

APPENDIX I: Provides the web documentation.

1. General Description

1.1 Overview

T4.3 (Scalable Policy-based Model Distribution from Cloud to Edge) and this deliverable (D4.13 - i4Q AI Models Distribution to the Edge) which summarizes the work done in the task are highly inter-related with T4.4 (Workload Placement and Deployment) and the corresponding deliverable (D4.6 - Edge Workloads Placement and Deployment). Thus, for completeness several of the sections are included in both deliverables. This mostly applies to background information providing details on the technological area and the challenges tackled by these complementary tasks. The specific implementation related sections are separated between both these deliverables.

Edge computing is establishing itself as the architecture of choice for many industries across many use cases. This trend is important as well for AI at the edge capabilities, especially when dealing with real-time AI related scenarios. There are several important reasons for this trend, such as processing data close to the source, thus reducing latency of the response, especially when working in real-time. In addition, network bandwidth is conserved, serving together as a first line of defence to tackle big data challenges by handling data locally as much as possible, rather than serve as a pipeline for all data to be transmitted to the cloud and be processed there. In such an architecture the vast majority of data is processed at the edge and only a sub-set, such as feedback related information, is shared with the cloud. Security and privacy are addressed well in such an environment by keeping and processing the information locally without being externalized and sent to the cloud or a central data centre. Such an architecture may prove to be beneficial also for abiding by data related regulations, such as restrictions on the data storage and processing locations. This architecture support also working in a disconnected mode. When working offline all data is processed locally and the sub-set of data that needs to be transmitted to the cloud can be buffered until network operations resume.

There are several unique features that are prominent in such as an environment, chief among these is the scale, heterogeneity of computing platforms, and support for a disconnected mode of operation. The end goal is to develop an efficient, large-scale hybrid cloud edge infrastructure for edge computing, supporting AI operations at the edge by handling AI models and workloads lifecycle management. As briefly mentioned above, there are several reasons and advantages for having such a hybrid edge computing environment.

- Reduce the amount of data sent to the cloud. Significant portion of the data processing can be performed at the edge, close to where the data is generated, thus dramatically reducing the amount of data that has to be sent to the cloud. This allows more data to be processed at a lower cost in less time.
- Ability to work while disconnected from the cloud. This is a big advantage in many use cases where communication with the cloud is unreliable, expensive, slow, or is not always available.
- Privacy and security. In certain cases, some of the data must remain local and cannot be sent to the cloud, for example, due to regulatory reasons. In other cases, users may want to minimize the data exchange with the cloud to reduce the security risks involved with data in motion.

- Faster reaction time. In several use cases a fast reaction is important and the ability to process the data on the edge and react immediately is important.

Applications involving big data and/or IoT are examples of applications that can greatly benefit from a hybrid edge computing environment. Edge computing brings a set of challenges, such as being able to work in a resource constrained environment, management and monitoring, being able to work in disconnected mode, heterogeneity of devices and more. We focus on some of the key challenges involved in creating a scalable, manageable, hybrid edge computing, concentrating on AI on the edge capabilities.

Edge computing adds another tier of data processing to cloud processing. We define an architecture for a multi-tier hybrid edge computing platform that can efficiently service multiple types of applications and use cases. The platform addresses issues such as the management of edge nodes, distribution of model updates, smart workload placement, and more.

Edge computing is becoming prevalent in many sectors, including smart manufacturing. Factories may start experimenting at a small scale but in large and distributed production environments such capabilities are becoming a requirement. To support this trend, we design the AI on the edge component in a scalable manner, to support distributed installations, while ensuring that small scale scenarios are supported as well.

Edge computing typically involves large-scale deployments on a wide range of heterogeneous devices. Managing the artifacts that need to run in such an infrastructure is complex and hinders the growth of this field. We intend to alleviate management of AI models and workloads in an edge environment, and making it easier to use than current practices.

Managing a large number of edge nodes is a key aspect of edge computing. Edge nodes are diverse and come in a variety of specifications and supported capabilities. However, a clear trend is to run containerized workloads on the edge as well. One of the obstacles in front of this vision is the limited support for easy deployment of AI ingredients on target edge machines. This is an issue being tackled in this task. We aim at reducing deployment complexity of AI models and workloads in edge scenarios. Such AI models tend to change based on incoming feedback from the running system. Thus, an extension of this task shall support the deployment of AI models based on updated information concerning the efficiency of the currently running model and available resources. Taking into account both the model running in real-time and the state of resources on the respective edge nodes. This forms a loop from deployment through feedback leading to re-training and re-deployment. AI models and workloads exhibit different lifecycle models. Workloads tend to be more stable with a longer lifecycle loop, while associated AI models tend to be updated at a higher rate, to improve its results based on updated feedback obtained from the field.

The potentially large number of edge nodes presents challenges in terms of configuration, management, and monitoring of the AI workloads and the environment. We design and implement a component which is responsible for providing a central control point for the management of AI workloads and models running on the edge. This central point will be accessible from the cloud and will enable the user to specify the rules that govern the location in which certain computations should take place.

This solution provides assisted capabilities for distributing AI models to the edge. It includes the synchronization of information and state between a cloud-based entity and the models placed at the edge to be used by AI workloads orchestrated at the edge. AI models have different life cycle behaviour and pattern than AI workloads and thus they require differential treatment, including the possibility to place them at correct locations at the edge, monitor their effectiveness, and be able to receive feedback that may lead to retraining and redistribution of refined models to the edge. A sub-set of these capabilities shall be demonstrated within the FACTOR use case (for more details please see subsection 3.2).

This task aims at alleviating the process of deployment of AI models on the edge by introducing flexible and scalable lifecycle management of AI on the edge components. AI at the edge enables AI computation logic close to the data source.

Edge platforms are expected to improve the efficiency of edge devices while removing the obstacles for developers and administrators managing the complex hybrid environment. Infrastructure for supporting edge applications based on a microservices approach is presented. The current envisioned use cases in the pilots may be rather small scale at the moment but we envision advanced features to cater for future large-scale scenarios which represent current and future directions and trends envisioned also in the smart manufacturing space. Once the technology is demonstrated on a single machine, requirements for scale are expected to follow soon, requiring support for more workloads, more machines, more models, more production lines, more plants, etc. Even if at a first step modest requirements are foreseen, we intend to build and use scalable technologies (such as ACM and K8s on the edge) to be present and future ready.

The main capabilities of this solution shall be demonstrated in conjunction with the Line Reconfiguration Tool (LRT) solution in the FACTOR pilot. Need for scale in this arena can be seen for example in the IBM-FORD joint effort on defect identification across many plants and devices. The solution was deployed at several plants and embedded in multiple inspection points per plant. As a consequence and aftermath of a small scale demonstration, the technology and deployment is expected to be expanded to additional plants and use cases.

In essence, the objective of this solution is to help deploy and operate AI models close to the source of the data feeding the models in action, so that, for example, production of defected parts is minimized, and overall effectiveness of the production machines is increased. It supports industrial companies in the quest for autonomous operation in manufacturing environments, taking advantage of AI. Consequently, the production rate can be increased, while waste and cost can be reduced. Thus, the main outcome of this solution is a model distribution component designed for high scale edge computing targeting manufacturing and industrial applications. The main challenges and characteristics are scalability, automation, flexibility, and ease of use model distribution mechanism. This capability should support integration with an overall AI model lifecycle management. Furthermore, integration and coordination with workload distribution is targeted to ensure workloads and their AI models meet at the edge.

1.2 Related Work

Multi-Cluster management is a key aspect of a cloud-edge architecture. Large scale deployments are driving this trend. One reason for the increased demand in scale is the evolution of edge computing, which is growing rapidly with more and more workloads moving or extended to the edge to benefit from efficient use of bandwidth, low latency, and enhanced privacy and security. This means that far edge use cases now include a large number of edge nodes hosting a variety of workloads. In addition, Kubernetes has established itself as the platform of choice for running applications and services. Kubernetes is thus replacing many other platforms (e.g., Docker, and OpenStack) due to the uniform management it provides across different environments (such as public cloud, private cloud, near edge, and far edge). In addition, reduced footprint clusters are being produced, enabling Kubernetes (K8s) clusters to be deployed in more environments. This includes far edge scenarios which, until recently, didn't seem suitable for K8s. Offerings such as k3s (https://k3s.io), take this approach to the extreme and offer K8s versions that can run on devices with very limited resources such as a Raspberry Pi with less than 1GB of memory. This opens up the door for using K8s as the underlying orchestrator in many edge use cases (such as smart manufacturing, and more). Large scale edge management frameworks are being explored. For example, Open Cluster Management (https://github.com/open-cluster-management) offers comprehensive multi-cluster management capabilities with extensions to address large scale far edge use cases.

1.3 Features

The main features provided by this solution include:

- Distribute AI models to the edge where they are expected to be used by locally running workloads. The AI model distribution is coordinated with the workload distribution mechanism to ensure that the right set of AI models is made available for the workloads that use them.
- Help deploy AI models at the edge in scale. To enable that, a policy-based deployment pattern is used via associating labels with the potentially different target nodes.
- Manages lifecycle of AI models, from creation at the cloud, initial deployment at the edge, and re-deployment when revised models are available. Finally, when not needed anymore the deletion of the model is supported as well.
- Support policy-based placement mechanism for AI models that eases the task of the administrator by enabling the specification of rules for eligible targets in a simplified manner.
- Support GitOps based mode of operation. The connection point between the model creator and the model deployer is performed via git. New resources that get pushed into git are retrieved to be deployed on the target nodes associated with the requested labels. Thus, the developer does not have to master deployment issues, but rather concentrate on their area of expertise and only needs to use Git in a standard manner and the automation behind this task will follow through with required deployment at the corresponding target nodes.

- Support the deployment of AI models with Red Hat Advanced Cluster Management for Kubernetes (RHACM). ACM serves as the basis for enabling deployment operations at scale. We provide the link between the creator and the deployer of the models.
- Support AI models wrapped in containers. In this mode, the models are ready to be loaded by the corresponding AI workload to be used internally.
- Support AI models wrapped in an HTTP server. Model deployed as a microservice exposing a REST interface that can be used by additional components such as the AI workload. Upon deployment, the model wrapper starts listening for incoming REST requests, and internally invoke the corresponding code.
- Operate seamlessly well for small to very large multi-site infrastructure common in smart manufacturing environments. Policies, labels, and rules can be simple and constantly grow to more complicated given the growing or shrinking deployment target space.
- Support lightweight orchestration engines such as k3s. Thus, the models can be deployed and made operational over a variety of host devices and architecture, from high to low footprint artefacts.

These features are demonstrated in the video showcasing release 1 capabilities.

2. Technical Specifications

2.1 Overview

Edge computing faces challenges in scale of supported target nodes, and heterogeneity of the underlying devices (different hardware, communication, processing power and so on). A clear trend is to run containerized workloads also on the edge. While many platforms still use container runtimes, such as Docker, Kubernetes is increasingly becoming the platform of choice for both the cloud and the edge nodes. As Kubernetes extends to the edge there is a growing demand to significantly increase the scale of multi-cluster management. This trend is expected to rapidly grow due to the combination of the following factors

- Kubernetes has established itself as the platform of choice for running applications and services. Kubernetes is thus replacing many other platforms (e.g., Docker, and OpenStack) due to the uniform management it provides across different environments (public cloud, private cloud, near edge, far edge).
- Edge computing is growing rapidly with more and more workloads moving or extended to the edge to benefit from efficient use of bandwidth, low latency, and enhanced privacy and security. This means that far edge use cases now include large numbers of edge nodes hosting a variety of models and workloads.
- Reduced footprint clusters are being offered which allow Kubernetes clusters to be deployed in more and more locations with varied resources. This includes far edge scenarios which, until recently, didn't seem suitable for Kubernetes. This opens up the door for using Kubernetes as the platform in many edge use cases.

We make use of the K8s support throughout the design, development, and deployment of these solutions.

Our research targets large scale edge platforms. An edge node can be anything from a tiny edge device to a multi-server edge cluster. We see a growing demand for such large-scale platforms across a variety of industries including telecom, manufacturing, automotive, industry 4.0, retail, IoT, and more. The scale of the platform mandates new and innovative solutions to a variety of management and automation aspects including lifecycle management for the edge infrastructure, and management of applications and services running on the edge. Special focus is given to Kubernetes edge clusters as Kubernetes becomes the platform of choice for both cloud and edge.

Our primary goal is to provide a scalable control plane capable of managing a large number of AI workloads and their associated models within Kubernetes edge clusters.

This task addresses the challenge of managing large scale Kubernetes based edge computing platforms on which AI models are deployed and executed. In our solutions, edge nodes are operated as Kubernetes clusters which can be low footprint single-node clusters or large multi-node clusters. A scalable control plane is designed and implemented to control and manage the models that run on these edge nodes. The control plane shall address the multiple management aspects that are required for such edge platforms. These aspects include lifecycle management of the Kubernetes clusters, lifecycle management of the workloads running on the edge nodes along with their models.

This task aims to provide a distributed resource orchestrator and manager to support the deployment and operation of AI models at the edge. Kubernetes is mainly used, along with ACM and potential extensions where needed, to orchestrate and manage efficiently available resources in local edge devices and across distributed clusters of edge devices. Deployment shall be pursued based on resource allocation policies. AI applications and models' lifecycle management are pursued.

This task serves as a central enabler for being able to analyze data at the edge, relying on low latency and high data thruput. Provided capabilities help to de-couple edge applications from the underlying infrastructure, thus enabling the placement and deployment on different target nodes, by using universal tools such as K8s.

This task addresses the challenge of orchestrating large-scale Kubernetes-based edge computing platforms and the associated AI models running within. A scalable control plane is designed and implemented to control the edge nodes and workloads that run on these edge nodes. The control plane addresses AI workload and associated models lifecycle management.

This AI on the edge component addresses the life cycle management of AI models used by the pilots and i4Q use cases. This component enables efficient placement and deployment of AI based workloads and their associated AI models, on the edge, close to the data source locations. It further provides automatic deployment, updates, and monitoring of workloads and AI models, rather than a manual approach. This is more suited for a large and dynamic environment in which the deployment-feedback-retraining-redeployment cycle should be supported in an automatic manner. The AI on the edge component provides centralized management of all the edge workloads and the AI models they are using. A single Control-Center, which may run in the cloud or on-premises, can control workloads across multiple machines, production-lines, and plants which significantly simplifies management tasks, reduces errors, and cuts down costs. The AI on the edge component is generic and can work with a large variety of workloads, AI models, and edge infrastructures. For example, the same mechanism can be used for an application that processes sensors data (heat, vibrations, etc.) and an application that processes high-resolution images for automated visual inspection. Thus, the component is not restricted to a specific solution but is rather a generic infrastructure mechanism that serves different components.

The objective of i4Q^{AI} is to address the management of AI-based models in a hybrid cloud-edge manufacturing environment. This capability is realized via the deployment of a multi-tier infrastructure covering the cloud as well as the edge. The main aim of i4Q^{AI} is to distribute AI models to the edge where they are expected to be used by locally running workloads. i4Q^{AI} ensures that the cloud and the edge are synchronized and respond together to the changing environment. The AI model distribution is coordinated with the workload distribution mechanism to ensure that the right set of AI models is made available for the workload that uses them.

i4Q^{AI} is designed to operate seamlessly well for small to very large multi-site infrastructure common in smart manufacturing environments. Thus, the scope may span from a small local plant to multiple sites organizations. This is achieved by employing a policy-based placement mechanism that eases the task of the administrator by enabling the specification of rules and labels for eligible targets in a simplified manner (such as the model type, model version, geographic area, or the existence of specific resources). Input consists of placement rules or

models to be distributed. Output consists of a running model that can be used by a corresponding workload, and an updated view of the state at the edge.

The architecture supports infrastructure running on the edge, being managed by a cloud component, to enable together the management of AI models across multiple locations in a dynamic environment. The intention is to enable such management at scale via a policy-based distribution mechanism. There is a tight coordination between the AI model deployment solution and the workload distribution mechanism.

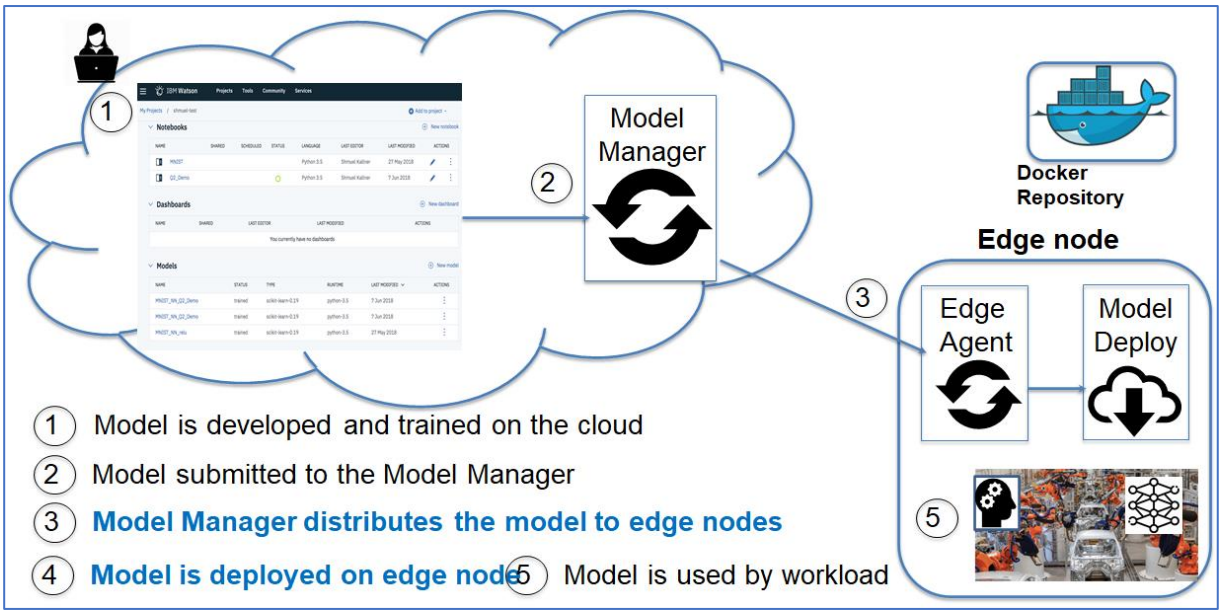

Figure 1. High level architecture and flow

Figure 1 depicts a high-level architectural view of the different relevant components, and provides a high-level flow of operations. At the cloud layer tools are provided for the development and training of the models. Once the developer is satisfied the model is handed off to the model manager component described in this deliverable. The hand-off is performed via GitOps. That component in turn interacts with a peer agent running on participating nodes. Once the manager has determined, automatically, the locations intended for the new or updated model, the corresponding edge agent is contacted. Through this interaction the edge node receives the model in question, either directly or through a link to git in which the model resides and can be retrieved. Once the model safely reaches its target, it is deployed and made available in the target node. The AI workload running on the same node now can proceed to use the newly deployed model. At a later stage, the model may be changed and re-trained in the cloud, causing a similar flow in which a new version of the model gets delivered and deployed on some target nodes. Local orchestrators take care of required local operations, such as rolling updates of the corresponding artifacts.

2.2 Architecture Diagram

Figure 2 presents a high-level architectural view of the AI on the edge solutions, and their interaction with their environment, at the different layers of operation. These components operate at the cloud level, the edge level, and the factory floor. At the cloud level resides the multi-cluster management component, which is in charge of managing all managed clusters in a scalable

manner. At this layer resides also the development and authoring tools for AI models and workloads. Once a new version of one of these components is ready to be deployed, the corresponding artifacts are pushed into a Git repository. The Git-repository serves as the contact point between the developers of the AI workloads and models and their respective edge deployment. Thus, the AI artifact developers can concentrate on their task and not worry about the artifacts' deployment and execution. From this point, in a GitOps manner, the Advanced Clusters Management (ACM) notifies the relevant managed clusters on which components need to be deployed, that new resources are available, which causes the managed cluster in turn to pull the corresponding Git resources from the corresponding Git repository. While consuming the new resources, the managed cluster, via the standard Kubernetes mechanisms pulls resources from the Container registry (AI workload and model containers) and activates the required containers for the AI workload and model. All these components reside and operate at the edge layer. At the factory level there are instrumented machines, equipped with IoT sensors and actuators, that continuously provide data to be analysed by the edge components of AI workloads and models. The workload may automatically control a machine and automatically apply configuration suggestions or can provide suggestions to an operator controlling the machine. Finally, data makes its way backwards as well from the machines at the factory level all the way to the AI models developers in the cloud. This feedback information is used to evaluate the efficiency of the model and to potentially retrain the model to obtain better results at the inference level. Once that is done the whole deployment cycle takes place as well culminating with a new version of the model being deployed at the chosen targets.

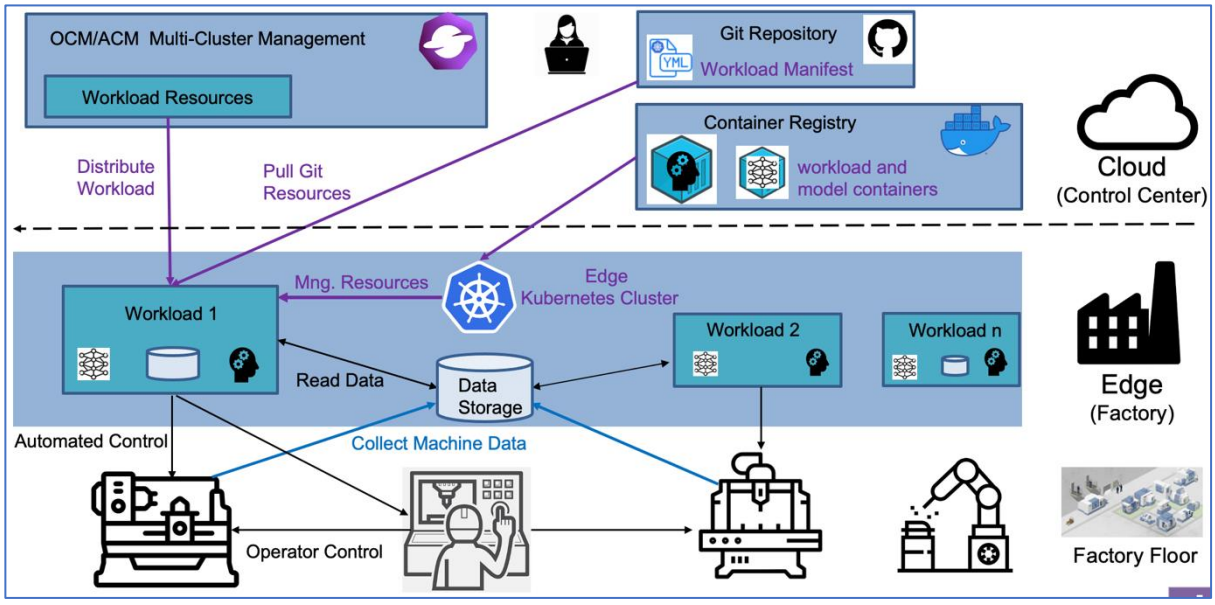

Figure 2. AI on the edge architecture

This solution includes different processes, which are mapped to the Platform Tier and to the Edge Tier of the i4Q Reference Architecture (see more details in D2.7). This solution shall operate at various levels of the architecture, a short analysis follows:

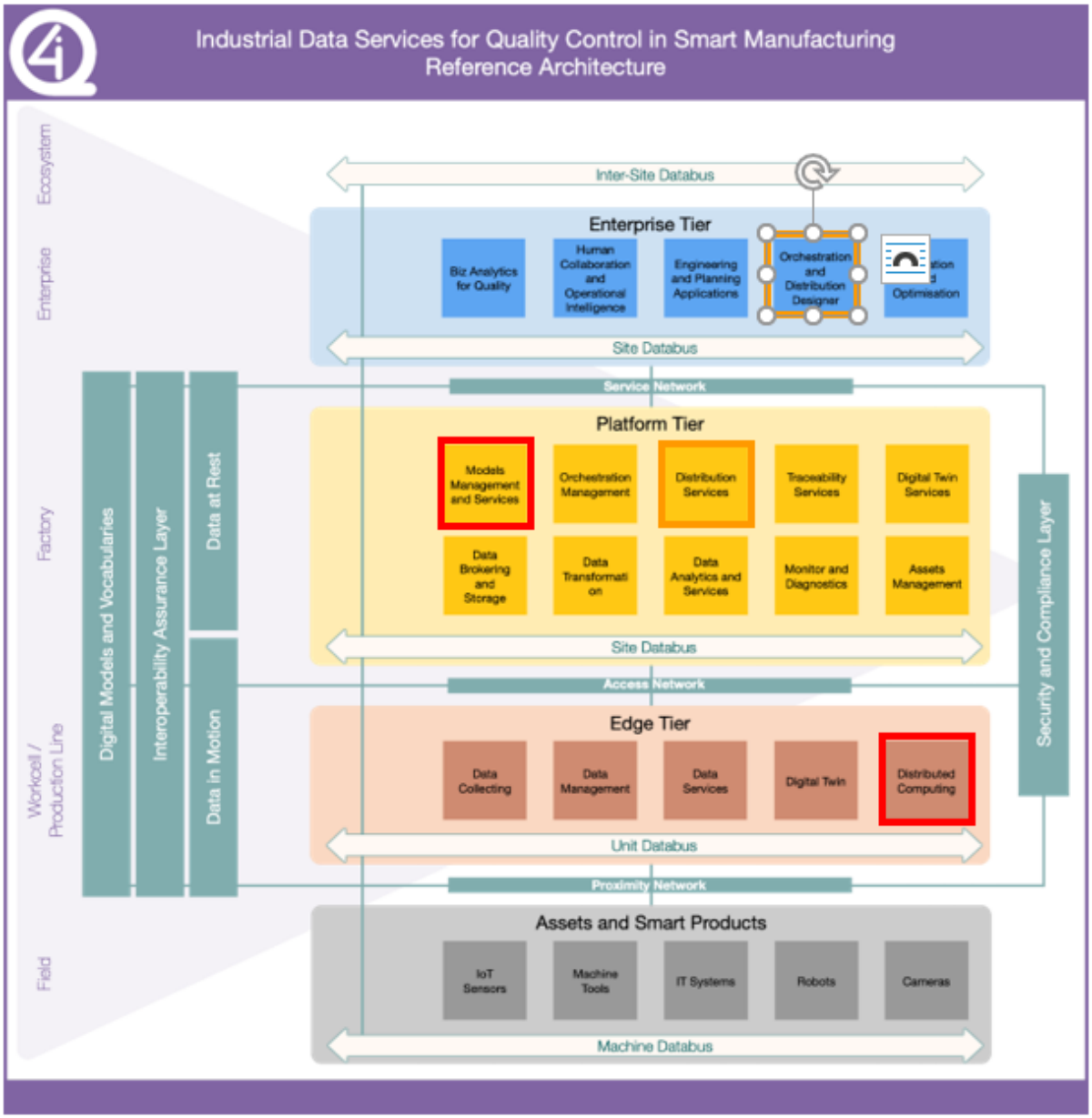

Figure 3. i4Q[RA] mapping with i4Q[AI]

Correspondence with the reference architecture:

- **Platform Tier:** "models management and services" will host the backbone for the realization of the capabilities of this solution. This shall serve as the hook between the enterprise tier authoring to the actual edge-based model deployment, while handling the corresponding specific lifecycle. Furthermore, this solution shall rely on the distribution service at the platform tier for placing models at the chosen edge locations. Will interface with the "orchestration and distribution designer" for creating and updating the edge models to be deployed (serves as an input to the model distribution solution).

- **Edge Tier:** The solution maps to the "Distributed Computing" subcomponent, enabling AI models handling on the edge. This component facilitates the deployment and execution of AI models on the edge. This enables processing at the edge and transmitting any required data back to the cloud, taking into account different conditions.
- **Enterprise Tier**: this solution relies on the orchestration and distribution designer to feed the initial model, handle incoming feedback, and provide updated versions of the model. When operating or encountering disconnected network mode of operation, between the edge and the cloud, the feedback loop from the edge to the cloud will slow down, which will cause a delay in retraining and re-creating a new version of the AI model to be deployed in the field. There is a need to take into account the velocity and heterogeneity of data to assess the communication required between the cloud and the edge.

2.3 Technology background

The main background technology we rely on is the Open Cluster Management (OCM) https://github.com/open-cluster-management (OCM is open source under Apache 2.0 License). This technology aims to ease the distributed control over various Kubernetes clusters. OCM uses a hub and spoke architecture for scalability. The hub serves as a control point for all managed clusters. The agent is installed in the managed cluster to manage locally operations directed to it from the hub. We rely on this technology and augment it to handle AI models and workloads.

Red Hat Advanced Cluster Management (RHACM) is the commercial product built on top of the open source OCM. ACM controls clusters and applications from a single console using policies across multiple clusters at scale. Managing K8s nodes and clusters at scale is the main capability provided. In addition, infrastructure for model lifecycle management is provided. We build on top of these capabilities features corresponding to models and AI workload lifecycle management. Deployment of applications based on policies and labels serves as an additional base capability w use and enhance to handle the artifacts coming out of this solution. The end result contributes capabilities for automating the enterprise workloads management at scale. ACM provides the basis for edge deployment on scale, thus helping to integrate edge computing with a cloud architecture and environment.

The technology we developed for scalable multi-cluster management (hub of hubs) serves as a basis for design at scale in a cloud - edge environment. The hub of hubs component is another example of enhancements added on top of ACM.

3. Implementation Status

3.1 Current implementation

The major goal of this task is to provide a scalable, flexible, and easy-to-use method for deploying AI models in an edge environment. The solution has been designed, developed, presented, and demonstrated to the consortium members, and the associated solutions and pilots. Current version relies on ACM and was demonstrated to work with various kinds of artifacts, using various kinds of orchestration engines, running on different HW configurations. Support for models wrapped in different interfaces and modes of work has been presented. The current capabilities have been mutually agreed upon with the LRT solution, which is the first solution to use this technology. The combined capabilities shall be used and demonstrated first in the FACTOR pilot. Currently considering further interactions for inclusion in the generic pilot.

The currently implemented capabilities are summarized below:

- The interaction between solutions providing AI models and this model deployment solution is performed via Git. New or revised AI models are pushed to a Git repository, which is monitored by the model lifecycle management and deployment solution. The information of the existence of a new model to be deployed is spread to the relevant managed clusters corresponding to the nodes that are potentially supposed to host the model in question. The relevant managed clusters than proceed to contact the corresponding git repository to grab the required resources, and deploy the corresponding models locally.
- This solution supports the deployment of models available in different flavors. Having the model wrapped in various formats, namely
- Having a model available as a binary object to be loaded at runtime by the corresponding AI workload. This option entails the workload to invoke model capabilities via its internal API.
- Having a model wrapped with a thin http server layer, employing a microservices approach, in which the workload interacts with the model via its exposed REST interface.
- Support different kinds of model packaging
- Model can be wrapped as a container, which is deployed by the system on the target edge device
- Model can be packaged as a container inside a helm chart which is then expanded and deployed on the target edge node. The model can be pulled from a container which in turn is pulled by a pod, or a storage mechanism mounted on a pod. The model in this option would still be packaged in a container, but deployed as a part of a helm chart and the container is mounted to a Kubernetes deployment or a pod.
- Once the capabilities have been demonstrated using a single target managed cluster, the demonstration has been expanded to support several managed clusters as target deployment options.
- This capability further demonstrates a flexible manner in which to select targets and deploy artifacts on different managed clusters, using policies and labels.

- Placement rules based on labels have been further demonstrated such that a change in labels causes a changed placement and target deployment edge node change.
- Further capabilities include the distribution of a model to a group of edge nodes, and another to a different group of edge nodes. Providing the flexibility to operate on multiple managed clusters simultaneously while providing the means and the freedom to interact with different subsets of available managed clusters based on the corresponding distribution policies.
- Demonstrate capabilities and interactions with the LRT solution to deploy their AI models on the target machines. LRT pushes new development to Git in which a listener is made aware of the new model to be deployed, and in turn notifies about a new version to be deployed. The latest development artifacts are than taken from Git and sent to be deployed on a corresponding edge node.
- Demonstrate the capabilities in the target FACTOR environment. The solution was demonstrated using a VM provided for the project within a FACTOR server. The manner in which deployment on the FACTOR environment was achieved has been demonstrated.
- Demonstrate different underlying orchestrators (K8s, K3s). Edge comes in different flavors thus we provide support for deploying and orchestrating AI models on various target infrastructure. The underlying orchestrator employed is a flavor of Kubernetes. We demonstrated the capability to deploy models on different machines, being strong and powerful or without ample resources.
- Use K3s or other low footprint standard K8s as the edge cluster. Specifically, K3s was used since nodes with limited resources are often used in such edge environments.
- Flexibly control separately model and workload. Models and Ai workloads may require differential mechanisms to handle since the lifecycle requirements are different; AI workloads tend to be more stable and long living which models tend to be re-trained and thus re-deployed on a smaller lifecycle time span.
- Capability to update model in some target nodes, update workload in other destinations. For example: 3 clusters placement, with differential deployment of workloads and models.
- Flexibility in distributing versions of the model and of the workload to different clusters. The deployment of different versions is made available via the labeling mechanism.
- Managed clusters individually download the necessary artifacts after being notified as to the existence of a new item to be deployed.
- End-to-end Demonstration: pack a model as a container or wrapped with an http server that provides access to the model. Model combined with inference engine, accessed through REST. Show dynamic, flexible, and scalable nature of the solution.

3.1.1 Sample UI

The UI used and exposed currently for this solution relies heavily on the ACM dashboard which is used as the underlying technology. Some samples are provided below:

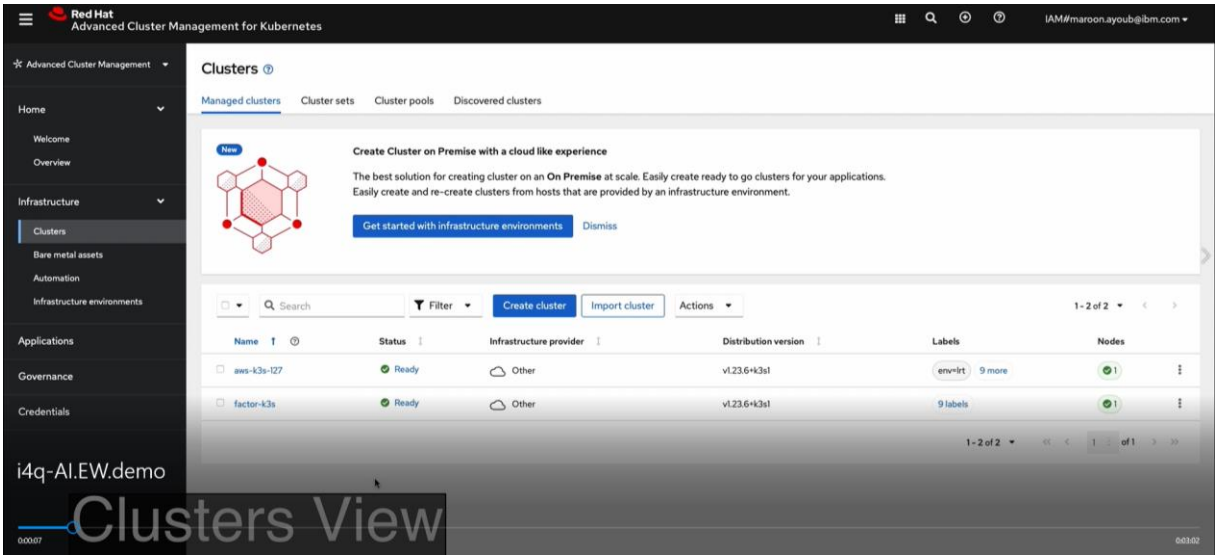

Figure 4. Clusters view

In **Figure 4** we can see a list of the clusters currently being managed by this instance. In this sample we can see two managed clusters listed corresponding to a cluster on the amazon cloud and another cluster in the FACTOR pilot environment. Additional actions are supported a swell such as creating a new cluster or importing an existing one.

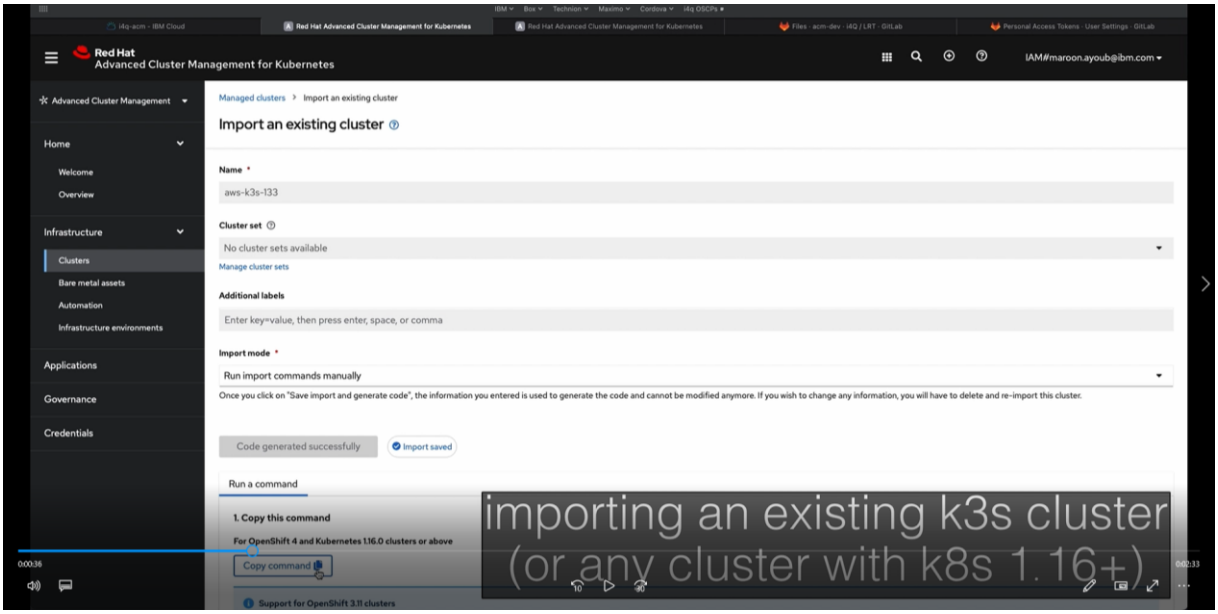

Figure 5. Importing an existing cluster

In **Figure 5** we can see the UI for importing a new cluster. In this case its' a cluster residing on the amazon cloud, supporting k3s, which is the low footprint version of Kubernetes.

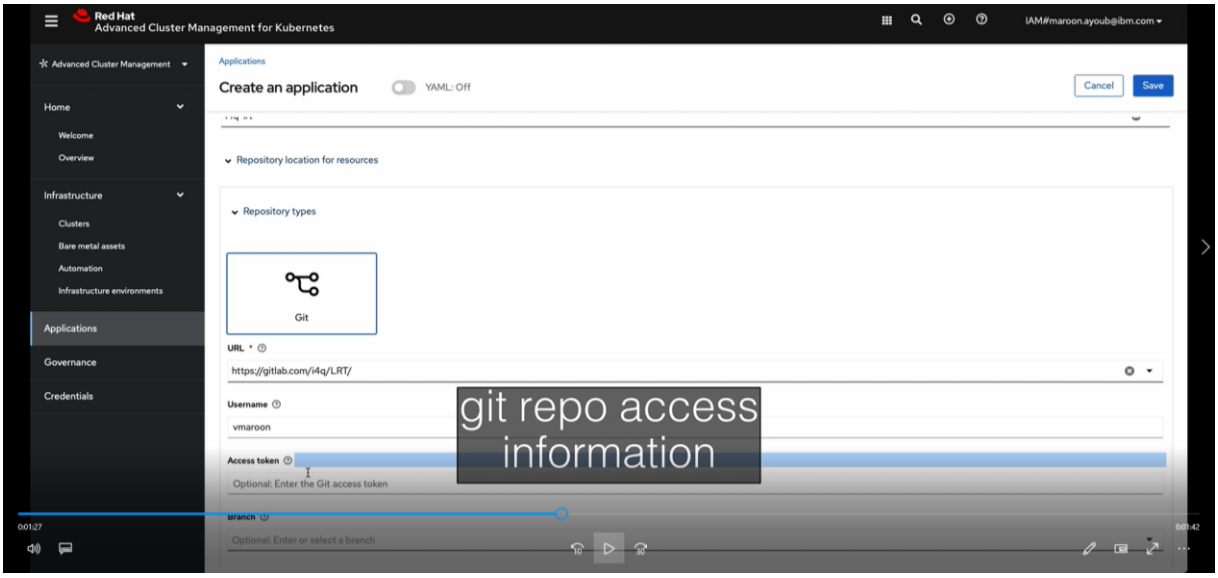

Figure 6. Application creation

In **Figure 6** we can see the application creation process, which points to the corresponding git repository where the application code resides.

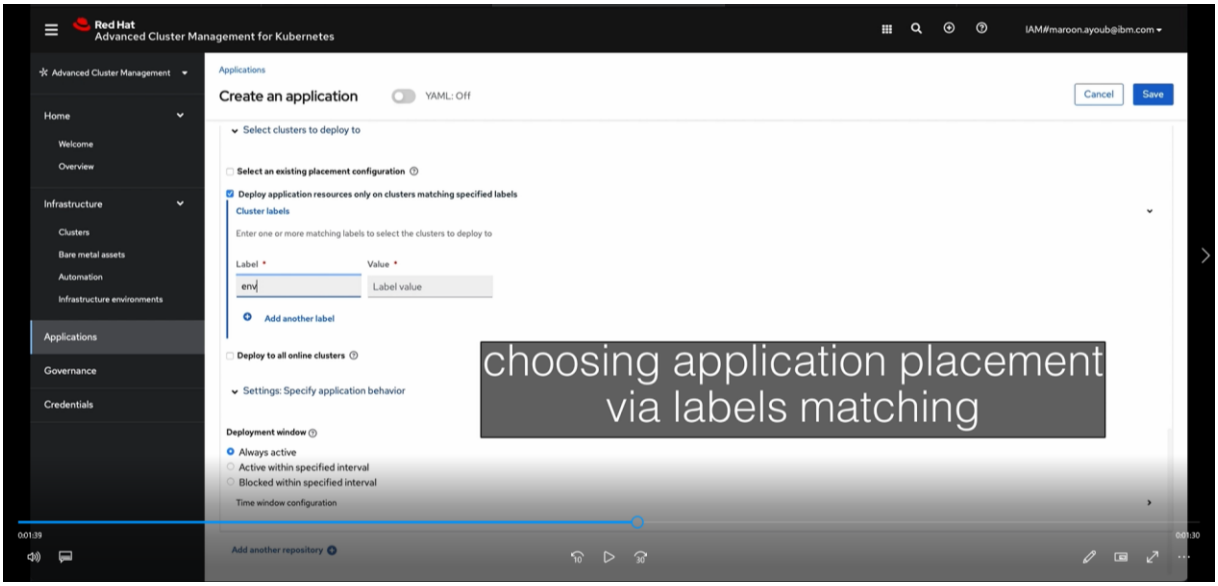

Figure 7. Adding labels

In **Figure 7** we can see one of the manners in which to associate labels with application such that the deployment process can later use these labels to assess adequate target managed clusters.

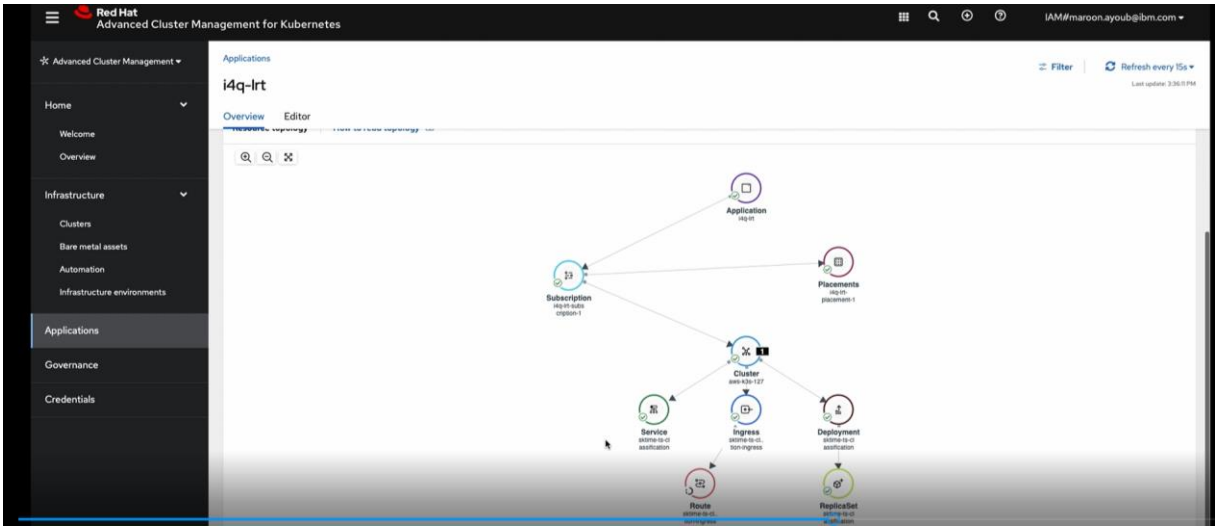

Figure 8. Application topology view

In **Figure 8** we can see the application topology view, which shows the application and the associated services and components.

3.1.2 Solution features mapping with user requirements

Requirements including the i4Q^{AI} solution are presented in the table below.

Entity	Requirement	i4Q^{AI} Features	Comments (Group of tasks)
Model generating solutions (such as LRT) that wish to deploy their models on the edge	Interface with different solutions	Expose interfaces and tools for deployment of AI models on the edge	Deploy-monitor-redeploy cycles.
Model generating solutions (such as LRT) that wish to deploy their models on the edge	AI models distribution	Policy based placement and deployment of generated AI models	Deploy-monitor-redeploy cycles.
FACTOR pilot	Low latency interaction between the sensorized machines and the AI models inference engine.	AI models deployment to the edge	Deploy-monitor-redeploy cycles

Table 1. Solution featured mapped to requirements

As can be seen in **Table 1** above, the requirements are mostly of the non-functional kind, even though some functional requirements are necessary to fulfil the requirement. The main requirements are geared towards the possibility to operate efficiently in real-time, while reducing

latency, enhancing data privacy and security as a whole, conserving network resources, while being able to operate also in a network disconnected mode. The function requirements, such as the support for flexible deployment of AI models in the edge are geared to fulfil such NFRs.

The consequence of the non-functional requirements drives the deployment of models at the edge, and the scale points to an automatic rules-based deployment mechanism.

3.2 Role in the corresponding pilot

This solution shall be demonstrated as a part of the FACTOR pilot. The FACTOR pilot focuses on improving the efficiency of the manufacturing process, mainly in two dimensions, namely, the quality of manufactured pieces (towards a zero defects production), and the overall efficiency of the production process. The main method to achieve this goal is to incorporate more digitization instrumentation and measurable automation into the process. The end goals are to reach the zero defects in manufacturing goal (P4_BP01), and to reduce the number of times a machine must be stopped, and the overall time machines are performing non-productive work (P4_BP02). To achieve these goals, quality control applications that monitor the production process and adjust it to achieve better efficiency are introduced. These applications typically employ AI/ML to analyze the collected data and decide on the best corresponding actions. Due to a variety of reasons, including the real-time nature of the control process, security concerns, and connectivity requirements, the quality control applications should run on the edge, i.e., in the factory close to where the data is being generated. Management of the edge applications and the AI/ML models they require for their operation is an important aspect of the overall tasks required to achieve the pilot's goals.

There is a good match between the intention of the pilot and the capabilities provided by the AI on the edge support suite. At this stage we shall look at cloud native technologies for easier installation, flexibility, and edge friendliness. A first i4Q solution to make use of the AI on the edge components is the production Line Reconfiguration Tool (LRT), being developed by UPV. AI on the edge components shall support various kinds of artifacts, such as enable workloads to dynamically load required models from shard storage or be wrapped in an http server as a microservice, deployed via helm charts, and so on. AI on the edge components shall cover a wide range of target scale, from full-fledged K8s / OpenShift environments to small scale such as k3s which is easily deployed, edge friendly, and can run on a single not necessarily strong VM.

The AI on the edge incorporation will help in some of the fundamental aspects of the pilot such as the collection of a large amount of data, serving as an enabler and input for gathering relevant parameters and interactions among them, as a consequence recommending config and parameters changes. At the background there is a continuous feedback loop from the edge to the cloud to enable producing enhanced AI models, which in turn need to be redeployed to the correct target locations. The AI on the edge component is a key pillar for digitation and automation aspects of the foreseen pilot. The edge component can be a part of a solution to improve the manufactured product inspection process; increasing the rate in which such inspections are performed, improving the accuracy of the inspection, as well as being capable of predicting ahead of time that the process is not moving in the right direction and attempt to alter it in real-time, while the process is running, to avoid defects in the manufactured items and reduce the rate of machine stops. These processes can save a lot of time and money for the manufacturing entity,

since some of the production processes are long and the raw materials used may be expensive as well. The AI on the edge can be an integral part of both the manufactured goods production and the whole production process improvement, focusing mainly on machine stops.

In the pilot, sensor data acquisition as close as possible to the CNC machine is supported. For example, the fluctuation in time of parameters such as tool temperature and power consumption is of interest, as it may prove to be a predictive measure. The AI on the edge components serve as an enabler for such an architecture. Another mode of interaction between the technology and the pilot calls for each time the machine is stopped, for the solution to take into account the cause of the stop which may cause readjustments to the model. This in turn will make use of the AI on the edge components for re-deploying the revised model to the right target nodes.

3.2.1 Demonstration

The current demonstration of this solution includes the following capabilities: the main goal of this demonstration is to illustrate the deployment of workloads and AI models to edge Kubernetes clusters or single nodes running a flavour of Kubernetes. The AI models are dynamically deployed on the edge and loaded by the corresponding workloads which makes use of the model to carry out their inference tasks. This solution supports different flavours of AI Models, such as wrapped in an HTTP server that runs as a micro-service and used by the corresponding workload using a REST interface. In addition, the model can be supplied as a binary object to be embedded in the workload which directly invokes the model interfaces. Differential lifecycle management of workloads and AI Models (update, configure, delete) is supported as well. In the cloud, the main management entity controlling all operations is based on ACM / OCM, and servers as a central management of all participating edge clusters. The underlying orchestrators used are based on Kubernetes and may run different variants based on the characteristics of the target nodes. We demonstrated the capabilities on K8s as well as K3s. In addition, Helm charts can be used to deploy multiple micro-services along with their corresponding AI Models.

All capabilities are shown at the recorded demonstration.

3.3 Future developments

Enclosed are several avenues which we intend to pursue for future releases, potentially after the official end of the project. These items need to be prioritized thus not all activities shall be implemented and demonstrated within the confines of one of the pilots. Some of the capabilities may be integrated and demonstrated via the generic pilot as well.

- Allow the edge cluster to run anywhere (e.g., in a target VM provisioned for the pilot, or in a container). Increasing the kinds of target nodes, we can manage uniformly.
- Bases on Git scale, a central hub may grab resources and deploy on managed cluster, or each managed cluster grabs directly from git. Currently this solution concentrates on having the managed clusters individually download the necessary artifacts. This is not necessarily the best solution for all environments. Thus, we aim to support different interaction mechanisms with Git such that the span of environments we can support is larger.

- Work with OCM rather than ACM – open source vs. the product version of it. Easier to have capabilities incorporated to the open-source version which serves as the pipeline for the upstream product.
- Scalability tests – see how many managed cluster and managed entities we can sustain. This shall be tested while deploying or simulating a large number of managed clusters.
- Demonstrate scaling up the initial demonstration environment to multiple sites. Currently we demonstrate this solution within a single FACTOR site, but the ultimate goal is to scale the deployment opportunities to uniformly control multiple distributed sites. As a first stage we demonstrate the solution working on different flavors of cloud and dedicated VMs.
- Connect to additional i4Q solutions and pilots - adjust to collaborate with additional partners. This capability may be pursued as a part of the generic pilot based on requirement and final content exposed by the generic pilot.
- End-to-end Demonstration: pack a model as a container or wrapped with an http server that provides access to the model. Model combined with inference engine.; accessed through REST. Show dynamic, flexible, and scalable nature of the solution.

3.4 History

Version	Release date	New features
V0.1	01/04/2022	Initial release, simple model deployment
V0.2	01/05/2022	Support multiple managed clusters as targets
V0.3	01/06/2022	Label based deployment
V0.4	30/06/2022	Integration in the FACTOR environment
V0.5	01/12/2022	Revised integrated demonstration
V1.0	30/12/2022	Final version

Table 2. History of versions

4. Conclusions

Capabilities, design, and interactions supported by an AI on the edge components were detailed and demonstrated. Such capabilities involve several professionals, for example developers, data scientists, developers, or administrators of edge platform vendors. Developers and administrators can make use of the AI on the edge at scale friendliness provided by the described capabilities. In turn this may attract edge platform providers, not only to make their platform more attractive to their customers, but also to enhance the utilisation of their underlying resources. Thus, Edge Software providers will enjoy enhanced application deployment and efficient life cycle management, edge-based AI application developers will be able to focus on the logic of their specific service and not worry about the heavy machinery at the infrastructure level that deals with placement and deployment issues in a complex target environment.

A first version of this technology is used in the FACTOR pilot, paving the way for additional scalable deployment targets. Later versions took a step towards distributed and scalable support by supporting simultaneously many targets and edge devices. In this deliverable we focused mainly on AI models deployment, while in D4.6 we operate in the same environment, while concentrating on AI workloads and applications life cycle management.

Appendix I

I4Q AI Models Distribution to the Edge (*i4Q*^{AI}***)*** *web documentation can be accessed online at: https://i4q.upv.es/13_i4Q_AI/index.html without putting the information in the deliverable as it is already in the web page*