ACCORDION

Adaptive edge/cloud compute and network continuum over a heterogeneous sparse edge infrastructure to support nextgen applications

Deliverable D6.4

Pilot Prototypes Implementation

ACCORDION receives funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 871793

DOCUMENT INFORMATION

PROJECT	
PROJECT ACRONYM	ACCORDION
PROJECT FULL NAME	Adaptive edge/cloud compute and network continuum over a heterogeneous sparse edge infrastructure to support nextgen applications
STARTING DATE	01/01/2020 (36 months)
ENDING DATE	31/12/2022
PROJECT WEBSITE	http://www.accordion-project.eu/
TOPIC	ICT-15-2019-2020 Cloud Computing
GRANT AGREEMENT N.	871793
COORDINATOR	CNR
DELIVERABLE INFORMATION	
WORKPACKAGE N. | TITLE	WP6 Integration, Pilot Implementation & Evaluation
WORKPACKAGE LEADER	AEGIS
DELIVERABLE N. | TITLE	D6.4 Pilot Prototypes Implementation
EDITOR	Michalis Vakalellis (AEGIS)
CONTRIBUTOR(S)	Maria Pateraki (OVR), Zbyszek Ledwoń (ORBK), Thomas Loven (PLEX), Michal Rubaj (BSOFT), Emanuele Carlini (CNR)
REVIEWER	Bartłomiej Lipa (BSOFT)
CONTRACTUAL DELIVERY DATE	06/2021
ACTUAL DELIVERY DATE	30/06/2021
VERSION	1.0
TYPE	Demonstrator
DISSEMINATION LEVEL	Public
TOTAL N. PAGES	44
KEYWORDS	Pilot prototypes, Proof of concept, Use cases implementation

EXECUTIVE SUMMARY

This deliverable describes the implementation and operation of real-life experiments from the virtual reality, multiplayer mobile gaming and QoE in digital content delivery domains in order to demonstrate how the ACCORDION solution has been applied in real-world environments. It provides a report on the progress of the operational experiments' specification; the preparation of the datasets and experimentation environment to be exploited in each pilot prototype; and the indicators to be measured in order to validate both the operational (business) and technical performance of the ACCORDION platform, according to the ACCORDION pilot plans and experimentation protocol. Furthermore, this deliverable provides the guidelines for the execution and evaluation of pilots, in order to serve as proof of concept for the effectiveness of ACCORDION offerings. This is the first out of two implementation cycles that will validate the promised impact delivery.

DISCLAIMER

ACCORDION (871793) is a H2020 ICT project funded by the European Commission.

ACCORDION establishes an opportunistic approach in bringing together edge resource/infrastructures (public clouds, on-premise infrastructures, telco resources, even end-devices) in pools defined in terms of latency, that can support NextGen application requirements. To mitigate the expectation that these pools will be "sparse", providing low availability guarantees, ACCORDION will intelligently orchestrate the compute & network continuum formed between edge and public clouds, using the latter as a capacitor. Deployment decisions will be taken also based on privacy, security, cost, time and resource type criteria.

This document contains information on ACCORDION core activities. Any reference to content in this document should clearly indicate the authors, source, organisation and publication date.

The document has been produced with the funding of the European Commission. The content of this publication is the sole responsibility of the ACCORDION Consortium and its experts, and it cannot be considered to reflect the views of the European Commission. The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content.

The European Union (EU) was established in accordance with the Treaty on the European Union (Maastricht). There are currently 27 members states of the European Union. It is based on the European Communities and the member states' cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice, and the Court of Auditors (http://europa.eu.int/).

REVISION HISTORY LOG

VERSION No.	DATE	AUTHOR(S)	SUMMARY OF CHANGES
0.1	30/4/2021	Michalis Vakalellis (AEGIS)	First ToC
0.2	7/5/2021	Michalis Vakalellis (AEGIS)	Revision
0.3	24/05/2021	Maria Pateraki (OVR) Zbyszek Ledwoń (ORBK) Thomas Loven (PLEX) Michal Rubaj (BSOFT) Emanuele Carlini (CNR)	1st round of Contributions
0.4	18/06/2021	Maria Pateraki (OVR) Zbyszek Ledwoń (ORBK) Thomas Loven (PLEX) Michal Rubaj (BSOFT)	2nd round of contributions
0.5	25/06/2021	Bartłomiej Lipa (BSOFT) Emanuele Carlini (CNR)	Revision
1.0 (Final)	30/06/2021	Michalis Vakalellis (AEGIS)	Finalization

GLOSSARY

AB	Application Bucket
AR	Augmented Reality
ASR	Application Status Registry
CCU	Concurrent Users
EU	European Union
EC	European Commission
GPU	Graphics Processing Unit
H2020	Horizon 2020 EU Framework Programme for Research and Innovation
KPI	Key Performance Indicator
LS	Local Service
QoE	Quality of Experience
QoS	Quality of Service
RID	Resource Identification and Discovery
RTT	Round-Trip-Time
VIM	Virtual Infrastructure Manager
VR	Virtual Reality
WP	Work Package

TABLE OF CONTENTS

1 Relevance to ACCORDION

1.1 Purpose of this document

This deliverable is the baseline of the first ACCORDION experimentation phase (M04-M19). In particular, it reports on the progress of the experiments' specifications as well as on the preparation of pilots to be executed and evaluated. It provides a description of the test environment, details of how operations are carried out within ACCORDION (sequence diagrams), and the interactions of involved ACCORDION components and paves the way for the execution and evaluation of ACCORDION use cases in the domain of i) Virtual Reality (OVR use case), ii) multiplier mobile gaming (ORBK use case) and iii) QoE in content delivery (PLEX use case).

In addition, it provides a detailed list of indicators to be measured and benchmarks to be used during the experiments in order to validate both the operational (business) and technical performance of the ACCORDION platform. Special emphasis is placed on ensuring the alignment of the operational experiments with the business objectives of the industrial partners expressed in the context of the defined use cases. Finally, it provides an initial list of use case assessment metrics and KPIs to be monitored in the evaluation phase, with respect to the expected project innovation and achievements. The operational experiments' specification reported in this deliverable will be further refined and adjusted during the evaluation phase to reflect data availability and platform functionality.

1.2 Relevance to project objectives

This deliverable realizes the ACCORDION efforts to address the needs of NextGen application to encompass frameworks for application development, the establishment of an adaptive and robust cloud/edge infrastructure continuum, and the abstraction of widely heterogeneous pools.

The pilot prototypes implementation fully supports collaborative VR, mobile and content delivery applications that are computationally- and data-intensive at the edge, latency- and QoE-sensitive, and scalable in nature. In this context, this deliverable is linked to the following ACCORDION objectives:

- Obj1: Maximize edge resource pool
- Obj2: Maximize robustness of cloud/edge compute continuum
- Obj3: Minimize overheads in migrating applications to cloud/edge federations
- Obj4: Realize NextGen application

1.3 Relation to other work packages

This deliverable provides a description of the experiments' implementation following the line of pilots planning (D6.1) and taking into consideration the progress of the work done from M10 to M16 concerning the advancements in ACCORDION system implementation (D6.3). The deliverable, also, consumes output

from WP2 regarding the user requirements (T2.1) and ACCORDION framework architecture (T2.3) and provides the guidelines for the execution and evaluation of the pilots (T6.6, D6.5). Finally, the pilot prototypes implementation aims to meet the expected use case KPIs and impact KPIs to be reported with in T7.3.

1.4 Structure of the document

This deliverable is organized as follows:

Section 2 reports on the pilot plans alignment based on the incremental and iterative nature of the ACCORDION framework applied to real-life experiments.

Section 3 provides a detailed description of the implemented pilots and the experiment workflows for each use case. It also describes the high-level interactions between the active objects of ACCORDION system for the execution of each use case and the components/features to be evaluated, the requirements to be met and roles that will be part of the execution and evaluation process.

Section 4 discusses the impact potential of ACCORDION framework in terms of KPIs to be fulfilled during the execution and evaluation of the experiments and finally, **section 5** concludes this document.

2 Pilot plans Alignment

2.1 Introduction

The experimental validation, focuses both on ACCORDION quality aspects (non-functional requirements) and at component and system level, according to the system functional requirements (described in D2.1). A main characteristic of the experimental process is that it considers both technical and business requirements. In particular, the definition of each experiment considers both aspects in an integrated way, aiming to validate not only the performance of the ACCORDION solution, but also its alignment with the needs of the use case owners. Thus, it requires the collaboration of both technology and business stakeholders in all phases. Another important aspect is that it is an iterative and incremental process, whereby the definition of the experiments is iteratively refined to an increasing level of certainty in order to reflect (a) revisions of the industrial use cases definitions and associated requirements and (b) progress in the design of the ACCORDION platform and associated technology characteristics.

This alignment process aims to ensure that the designed experiments reflect the actual business and technical requirements.

2.2 System, Requirements and Pilot alignment

This section provides a brief summary of the actions undertaken in the leadup to the operations phase of ACCORDION. Leading up to the MS4 and the 1st version of the system, the pilots have taken a number of steps and measures to prepare for the first implementation and evaluation cycle for the MS5 attainment regarding the 1st pilot evaluation (M19).

From the very beginning of the ACCORDION project, WP2: Requirements & System Design and WP6: Integration, Pilot Implementation & Evaluation all partners together the use case providers have closely collaborated to ensure the necessary interlinkages between the technical WPs and modules developed to ensure the best possible preparation for the start of the operational phase of ACCORDION.

Each release of the ACCORDION platform will be followed by a so-called pilot testing cycle (Figure below) featuring four important steps:

- Release of ACCORDION
- User Involvement & Testing,
- Evaluating, and
- Consolidating

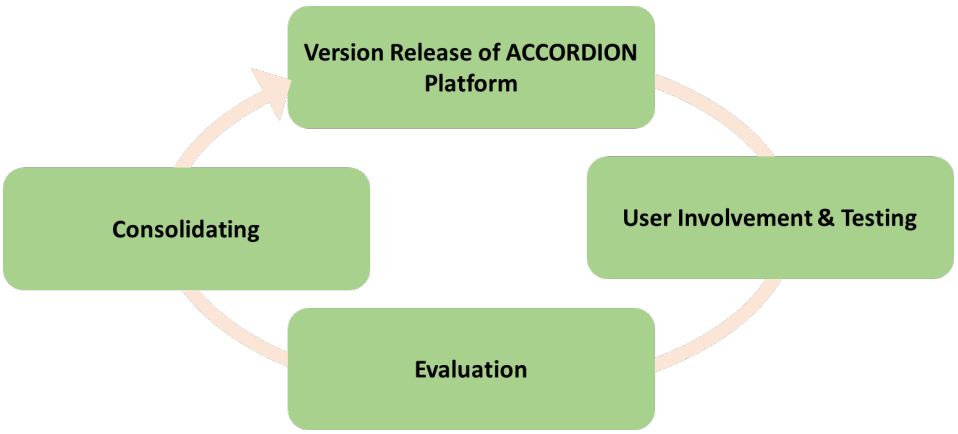

Figure 2-1: Pilot Testing Cycles following each ACCORDION Release

The release of ACCORDION is followed immediately by a phase that involves pilot users of ACCORDION and tests the release with them for functionality and usability, among others. The testing phase with user groups of the ACCORDION is followed by the evaluation phase.

In this phase, feedback from the second stage (user involvement & testing) will be aggregated following a clear feedback with recommendations to "Fix, Add, Improve or Remove" features of the ACCORDION and will serve specifically the technical partners with critical feedback for the development of the subsequent release. This allows for feedback to be translated into the next release of ACCORDION. The evaluation phase is followed by a consolidation phase which includes a lightweight process to confirm the findings from the evaluation. As part of this last step of the testing cycle, all feedback will be consolidated in the form of the Pilot Execution & Evaluation (D6.5, D6.10).

In specific, the anticipated feedback from the users is categorized as follows:

Add: Features and functionalities which have been identified by the user groups during the testing and feedback phase as relevant for a successful ACCORDION framework, but are not yet part of the release or the roadmap for upcoming roadmaps and therefore should be added.

Remove: Features and functionalities of the respective release which have been identified by the user groups during the testing and feedback phase as irrelevant or unnecessary - contrary to previous assumptions - and therefore should be removed. However, we do not expect to remove any functionality, at least at the first evaluation cycle.

Improve: Features and functionalities which have been identified by the user groups during the testing and feedback phase as unclear, confusing or otherwise insufficient. These features are nevertheless critical for the success of the ACCORDION and therefore need to be improved according to user needs.

Fix: Available features and functionalities which have been identified by the user groups during the testing and feedback phase as broken / not working, but are considered relevant for the ACCORDION and therefore need to be fixed.

In total, the ACCORDION project has foreseen two testing cycles starting at month 16, with the vision of providing a full functional platform after the second cycle of pilots' evaluation at month 33.

At the current status, the ACCORDION components offer a basic set of functionalities which is planned to build upon in the next iterations of the project, taking into account the feedback from the users. The three main core components of ACCORDION framework that are already available to be tested are:

- **Application Management Framework** that contains the methodologies and procedures to assist application developers working with ACCORDION, by hosting the application images and application models.
- **Edge/Cloud Continuum Management Framework** that implements the Orchestrator component that decides where each application will be deployed.
- **Edge Infrastructure Pool Framework** that manages the Miniclouds and the pool of available edge resources via a series of dedicated services that run on the edge.

Based on the above, the identified scenarios per pilot to be tested will follow the following plan:

Table 2-1: Scenarios per Use Case

Pilot prototype	Scenario	Supported in the first cycle	Supported in the second cycle
Collaborative VR module (Medical Sample App)	Scenario 0: Submission of the application	✓	✓
	Scenario 1.1: Start application	✓	✓
	Scenario 1.2: Request session	✓	✓
	Scenario 1.3. Run-time adaptation of the application		✓
Mobile multiplayer game by ORBK	Scenario 0: Submission of the application	✓	✓
	Scenario 2.1: Start application	✓	✓
	Scenario 2.2: Runtime adaptation		✓
QoE optimization in content delivery integrating Traqus /Anblick products	Scenario 0: Submission of the application	✓	✓
	Scenario 3.1.1: Start Edge Service Application	✓	✓
	Scenario 3.1.2: Run time adaptation		✓
	Scenario 3.2.1: User launches mobile Application	✓	✓
	Scenario 3.2.2: Multiplayer AR Game - Runtime Adaptation of Application		✓
	Scenario 3.2.3: User Device Hosts Game Session		✓
	Scenario 3.2.4: Multiplayer Session		✓

The evaluation process of the use cases also highlights the validation of the challenges and the subsequent enablers that ultimately signify the achievement of the whole ACCORDION vision. The following table

summarizes the implementation status of the ACCORDION platform that realizes the objectives 1-3 and their enablers, regarding the first implementation cycle.

Table 2-2: Objectives 1-3 and their enablers

		Yes	No	Partly
Objective 1: *Maximize edge resource pool size*	*Resource monitoring & characterization*			✓
	Resource indexing & discovery			✓
	Hybrid elasticity		✓	
Objective 2: *Maximize robustness of cloud/edge compute continuum*	*Intelligent, adaptive resource orchestration*		✓	
	AI-based network orchestration		✓	
	Resilience across the dynamic continuum	✓		
	Security across the dynamic continuum			✓
	Privacy-preservation across the dynamic continuum		✓	
Objective 3: *Minimize overheads in migrating applications to cloud/edge federations*	*Application data services model*			✓
	QoE assessment models		✓	
	DevOps to orchestrate network paths at the edge		✓	
	DevOps to support application deployment			✓

Below, for each enabler that is partly cover in the first evaluation cycle, there is a brief description of the current status of the relative component, elaborating on the challenges to be met during the second implementation cycle. In specific:

Resource monitoring & characterization: The Resource monitoring component is able to retrieve node and pod related metrics from Prometheus, the monitoring API component collects metrics and creates json files with a set of metrics. There is a Resource Monitoring & characterization instance in each MiniCloud, so each time monitoring API collects metrics or characterization information, it is for the MiniCloud that has been deployed to. Prometheus collects metrics every 30 seconds monitoring API generates and sends in a predefined frequency json files to the components that need specific metrics. The Monitoring API collects characterization information related to the nodes of the cluster. Characterization system can identify if a node is a VM, a RPi or a PC, along with its role in the K3s cluster and also contains information about the hardware of the nodes.

Future work on the component includes the resolution of the question: “isPrometheus able to produce metrics for VMs from the Kubevirt Framework”. In any other case, the Resource monitoring & characterization component can collect and produce sets of metrics for pods and nodes based on the needs of the monitoring consumers.

Resource indexing & discovery: The resource indexing & discovery is partly covered due to the amount of resource types available in the first iteration of the ACCORDION platform.

Security across the dynamic continuum: The main objectives of Security across the dynamic continuum are:

- Identify and assess technologies and tools applicable to the Edge/Cloud continuum framework and to be introduced as DevSecOps techniques to improve the security of CI/CD pipelines in a Secure by Design approach.
- Set up a secret management service to safely handle confidential material in ACCORDION environment, such as private keys, configuration passwords etc.

The first objective has been partly covered by implementing a Static Code Analysis Service (SonarQube platform) available to all ACCORDION partners. In the next phase we will address the remaining part of the first objective by implementing a Container Security service and the second objective by implementing a Secret Management service using open-source software.

Application data services model: The application model focuses on the description of the deployment and termination actions that the orchestrator should perform based on the scenarios of the Use Cases. It employs an extended version of the TOSCA syntax in which it describes the orchestrator with its actions along with a set of application components with their properties, relationships and dependencies.

The Orchestrator and Lifecycle Manager use an internal component called “Converter” that is tasked with the parsing of an intermediate version of the application model called “intermediate model”. The latter is maintained in the Application Bucket and it is used to perform actions on specific running instances of the application components. For the purposes of the Orchestrator, the Converter, is able to convert the intermediate model to K3s configuration files for the case of deployment and termination. In addition, it can produce the configuration of a Load Balancer when a component needs an external IP.

The current version of the application model supports the definition of the order of actions that the application requires to be executed on ACCORDION. Further developments are expected to take place so as to allow the model to define more explicit relationships between the application workflows and the ACCORDION-based workflows.

DevOps to support application deployment: It was partially achieved by defining and designing the DevOps process based on the requirements of the individual use case owners. State of the art of DevOps solutions have been studied and some of the opensource solution have chosen for implementation in the Accordion deployment process. Structure and communication of the components involved in this process was developed as part of the project work.

For the realization of NextGen applications (Objective 4), please check sections 3.6.4, 3.7.4, and 3.8.4 for each use case respectively.

Subsequently, the association of the core functionalities and aspects of the platform that would be evaluated as part of individual subtopics; the high-level objectives of the project; the prevailing technical/scientific,

functional and non-functional requirements and a suggested list on the type of evaluation metrics/procedure for each subtopic are mapped to the individual implementation and evaluation cycles as summarized in the following table.

Table 2-3: Association of evaluation subtopics, objectives, and experiments/metrics

Subtopic description	ACCORDION Objectives	Metric/evaluation	Cycle I	Cycle II
Changes and complexity when migrating the application components to ACCORDION	Objective 3 Objective 4	User acceptance tests	✓	✓
Automated lifecycle management Service monitoring	Objective 2	Latency and RTT Traceability of components through-out their lifecycle. Detailed measurements by the platform		✓
Fault Tolerance	Objective 2	Detailed measurements by the platform	✓	✓
Resource discovery provision and service elasticity	Objective 1	Measurements on types of resources discovered and migration times Detailed measurements by the platform	✓	✓
Storage at the edge	Objective 3	Resilience, end-to-end availability	✓	✓
Service elasticity	Objective 1	Service migration time and migration cost Detailed measurements by the platform		✓
Deployment mechanism	Objective 3	User acceptance and detailed measurements by the platform	✓	✓
Definition of accepted QoE levels	Objective 3	QoE model based on standards Subjective tests		✓
Security and privacy	Objective 2	Security Policy (ISO 27001) Compliance/Adherence and detailed measurements by the platform	✓	✓
Latency, data rate, number of users	Objective 4	Measured in an end-to-end manner following different metrics	✓	✓

3 Pilot prototypes Implementation and Operation

3.1 Overview

3.2 General & Technical requirements

This section provides the current status of the General-purpose requirements and Technical/Scientific requirements. The general requirements are key features of the NextGen applications that are common through all three use cases, while the technical/Scientific requirements are mainly of technical/scientific nature and they are deriving from the overall project's vision to meet the needs of NextGen application providers.

Table 3-1: General & Technical Requirements completed in 1st Cycle

ID	Description	Req. level	Status	Comment
R_GP_01	ADMINISTRATOR: Login panel with user roles	Must	DONE	Implemented with Gitlab OAuth2.0 protocol. User detailed information and its roles come from Gitlab API as well.
R_GP_02	ADMINISTRATOR: User/Admin Dashboards	Must	PARTIALLY DONE	Initial version of dashboard was implemented
R_TS_13	APPLICATION PROVIDER: Validation pipeline ensuring application and deployment process correctness	Must	PARTIALLY DONE	A pipeline for testing was created, the build & deployment pipeline are being developed
R_TS_15	ADMINISTRATOR: A system showing particular deployment steps that failed	Should	PARTIALLY DONE	Has been implemented, though some amendments are still to be done

3.3 ACCORDION Platform Components

In the following, the platform components and their basic operational characteristics are described.

Life cycle manager: Life cycle manager component is the responsible to manage the application inside ACCORDION. Specifically, the lifecycle manager (LM) matches the application workspace name from event-bus to a procedure that is called **recipe.** The recipe is the way ACCORDION implements a workflow given by

the application. Then, the LM requests the recipes that ACCORDION needs to perform based on the selected application workflow to the application bucket.

Once the Application Bucket returns the appropriate application model and recipe to the LM, the latter together with Application Status Registry (ASR) process it in order to "understand" whether it involves existing application component instances or there is a need for deployment of new ones. In any case, the LM queries the Application Status Registry (ASR) for the application component instances that better meet a certain criterion.

Once the LM recovers the ids of the application component instances that it needs, it forwards them along with the recipe (or a variation of it) to the Orchestrator.

Orchestrator: The Intelligent Orchestrator (IO) component is the heart of the ACCORDION platform. The role of the IO is to manage all types of resources of the ACCORDION Federation, exploiting and filtering at a certain level of aggregation the info it receives from the monitoring sources, to allow the allocation of them to the various application components, considering the applications' QoS/QoE requirements.

It is responsible to deploy/undeploy components of applications over the set of EdgeMiniclouds (EMs) composing the ACCORDION Federation. Each EM is identified by a unique identifier, i.e., Minicloud1, and is managed by a Virtualized Infrastructure Manager (VIM).

The IO Is invisible for the ACCORDION Platform clients. It has four different REST interfaces with other ACCORDION components:

- LifeCycle Manager: through this interface, the IO receives deploy/undeploy commands and the set of components, belonging to the same application, to be deployed/undeployed. It is a server-side interface for the IO, composed of a single Endpoint.
- Resource Indexing & Discovery (RID): through this interface, the IO can submit query about the status of resources of nodes of the ACCORDION Federation. It is a mediated point of access to the characterization and monitoring components, which are not directly accessed by the IO. It is a client-side interface for the IO.
- VIM: through this interface, IO can submit to a single specific EM the set of components to be deployed on it, with the necessary K3S Yaml configuration file, provided by the Converter component, to realize orchestration on Kubernetes. Possible errors and component instances are provided by the VIM as returned values. It is a client-side interface for the IO.
- ASR: through this interface, IO can register the new deployed components or unregister the undeployed ones, submitting the component instance name provided by the VIM. It is a client-side interface for the IO.

The IO exploits an internal model, developed using MILP techniques and a solver, to realize the orchestration of components over the resources retrieved from the RID. This internal process produces a deployment plan that IO will use to know on which EM the various components should be deployed. The component is built in Python using Flask and Connexion to implement the REST OpenApi3.0 interface, and the CBC solver is used to solve the MILP optimization problem for the orchestration.

Application Bucket: Application Bucket (AB) is the technical component responsible for storing application images, and Application Model. It is invisible for the ACCORDION Platform clients. This component provides REST API, which enables the creation, edition and getting data. DevOps process will administrate stored data, and the Orchestrator will get it.

The AB was built base on the Spring Boot framework and use the non-relational database Mongo DB.

Application Status Registry (ASR): The Application Status Registry (ASR) is the component that maintains the state of the running applications and their underlying running instances (application component instances). At all times, it may be queried through a RESTful API so as to provide details about the application components instances based on their ID or namespace.

Resource Identification and Discovery (RID): The RID keeps a distributed index of the status of the resources of the minicloud in the ACCORDION Federation. The information on the RID are periodically refreshed by the monitoring agent at each minicloud. There is an instance of the RID on each minicloud. Each RID instance can be queried to obtain the resources for the whole federation or by specifying several filters.

Virtual Infrastructure Manager (VIM): The VIM handles a set of edge resources and infrastructures (the infrastructure pool), typically owned by a single provider, and offers the Minicloud API to provision and manage virtual resources layered on the infrastructure pool, and to deploy applications using those resources. VIM is able to manage heterogeneous infrastructures, i.e. both AMD/x86 and Raspberry (ARM) machines, and can provide different types of virtual compute resources: VMs, containers, unikernels; but also storage and network.

Resource monitoring & characterization: The Resource monitoring & characterization component collects metrics and information related to pods and nodes of a specific MiniCloud. It generates json files with different set of metrics based on the needs of the monitoring consumers. The ACCORDION components that consume information from Resource monitoring & characterization are:

- The Storage component receives storage related metrics
- The RID component receives node level characterization information and node related metrics
- The ASR component receives information related to the pods of a MiniCloud
- The Resilience Policy component receives namespace and pod related metrics

3.4 Terminology and Concepts

For the sake of readability, a new terminology is adopted which is highlighted throughout the text. In particular, the terms “application workflow”, “infrastructure-level operation (i-operation)”, “recipe” are introduced.

An application is distinguished between:

- the application itself, i.e., the notion of a black box implementing a certain functionality. E.g., the OVR, ORBK or PLEX application

- the components that comprise the application which are referred to as “application components”, i.e., the conceptual components that are implemented by the images of the application
- the application component instances, i.e., the deployed component images
- the application instance, i.e., the conceptual group under which a set of dependent or co-owned components belong to.

Furthermore, it is to be noted that the interaction between components and the message structure is described, excluding the actual interfaces (REST or other).

Application workflow: An application workflow is triggered through the client’s software, applications components, or the ACCORDION portal (dashboard). Both components interact with the rest of the ACCORDION platform through the App2ACC message broker. The command for the workflow contains:

- the id of the authenticated user (application provider) who issues the command
- the application to which the user is referring to
- the application workflow to be implemented
- (optionally) the application instance id
- the QoE criteria/requirements

An example of such a message is the following:

```
{
        "user":"u1234",
        "application":"OVR",
        “version”: “34.23”
        "workflow":"startApp",
        "application-instance-id":"accordion-ovr-1-0-0-123"      //nullable
        "criteria":{                        //anyOf
                        "geospatial":{          //nullable
                        "lat":123.4,
                        "lon":456.5,
                        "range":20
                },

        "latency":"13",          //nullable

}
}
```

Recipe and infrastructure-level operations: The recipe is the way ACCORDION implements a workflow given by the application. More precisely, the recipe is translated into a set of infrastructure-level operations (i-operation). For example, the example “*startApp*” command is matched to the respective set of i-operations [*find minicloud based on location, find appropriate resources, deploy component A*].

The recipe is requested with a message in the form:

```
{
```

```
        "application":"OVR",
        " recipe":"startApp"
}
```

Example

Workflow: If a component with these characteristics exists return its endpoint. Otherwise deploy it and returns the endpoint. Reference UC: OVR, ORBK, PLEX

Recipe: [asr_rank, orchestrator_deploy, wait_result_deployment, return_endpoint]

I-operations:

- asr_rank: Implemented by the ASR and QoE. Rank the currently running components of the given type according to a set of criterium. Return the first X components.
- orchestrator_deploy: Implemented by the orchestrator. Deploy the component according to the set of criterium. Return whether the deployment has been started.
- wait_result_deployment: Implemented by the ASR. Wait for a component to be running.
- return_endpoint: Implemented by the LM. Return relevant information to the original caller.

3.5 Platform Component Deployment

For the scenarios relevant sequence diagrams are presented and described in the following section including both application and platform specific components and their workflows. As the component deployment is common for all UC scenarios a base diagram is described, which is exploited across all UC scenarios.

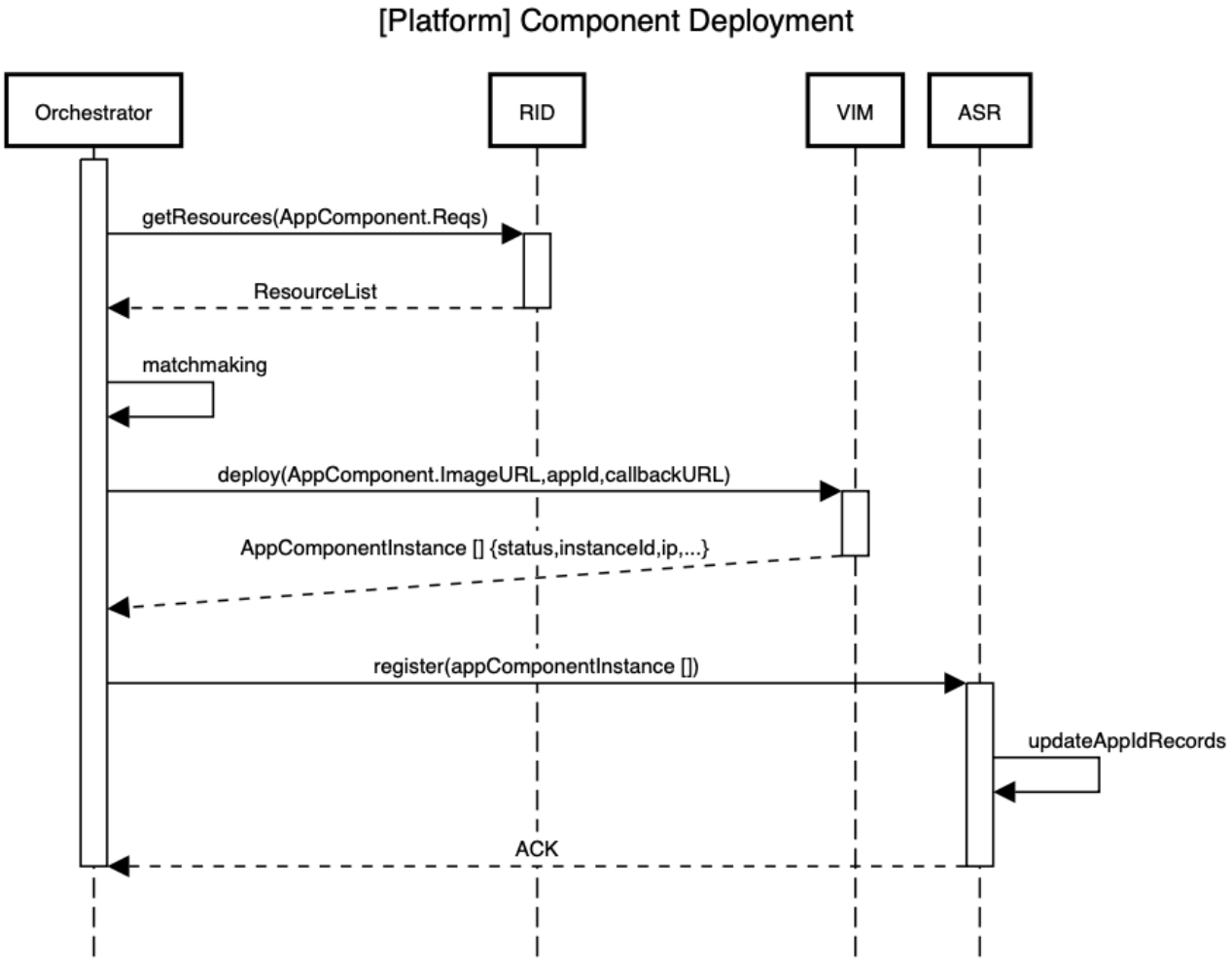

Figure 3-1: Platform Component Deployment

The deployment of a component in a minicloud is depicted in the Figure XX. In the ACCORDION Platform, the Orchestrator oversees managing the deployment process. The first step done by the Orchestrator is to contact the RID to get up-to-date information about the status of all the minicloud in the ACCORDION Federation. With this information, the Orchestrator chooses the best minicloud by performing a matchmaking algorithm. Then, it contacts the VIM on the minicloud with the relevant information for the deployment. The VIM replies when the request is accepted (it is worth mentioning that the information if the deployment is successful is communicated asynchronously by the monitoring to the ASR in a later moment). Finally, the orchestrator registers to the ASR the new request submitted.

3.6 Use Case #1: Collaborative VR

3.6.1 Pilot Prototype description

The application provided is a collaborative VR training module for mobile HMDs, which makes use of the platform to provide high-quality rendering via full application offloading, as well as a fault tolerant online cooperative session part. The application consists of three components:

- The application running on the edge (LS part)
- The signalling server
- The Relay server

The LS part is a Unity application exported for high-end (non-mobile) HMDs, for Windows. It contains the training module content, and can either be played offline, or online in collaborative mode (if a relay server exists). In order to start streaming to the end-user (mobile HMD), the LS part is instantiated along with a signalling server instance. It uses the Render Streaming Unity package on top of MAGES SDK to connect to the signalling server and establish a direct communication channel with the mobile HMD.

The signalling server enables the LS part and the mobile HMD to communicate via web sockets, and coordinates the application component and the mobile HMD according to WebRTC's signalling protocol. It is a NodeJS application which runs on a Linux container.

The Relay server is a Windows VM image which encapsulates Photon's On-premise relay server, which is used by multiple LS parts to enable online cooperative sessions between clients. Any client can either host, or join a session on a given relay server. It also enables host migration, in case the host client disconnects, allowing for the session to continue, with a new client as the host.

The LS part and the relay server are packaged as VM images, and compressed using XZ. This is required as the host operating system is Linux, and it does not support Windows containers.

3.6.2 Scenarios

As part of D6.1 four scenarios were described. Three would be supported in Cycle I, whereas the fourth one "Runtime adaptation of the application" would be part of Cycle II, considering that the QoE model will be available at a later stage of the project. Furthermore, Scenario 1.2 was renamed to "Request session" to address both create and join session option under a single scenario.

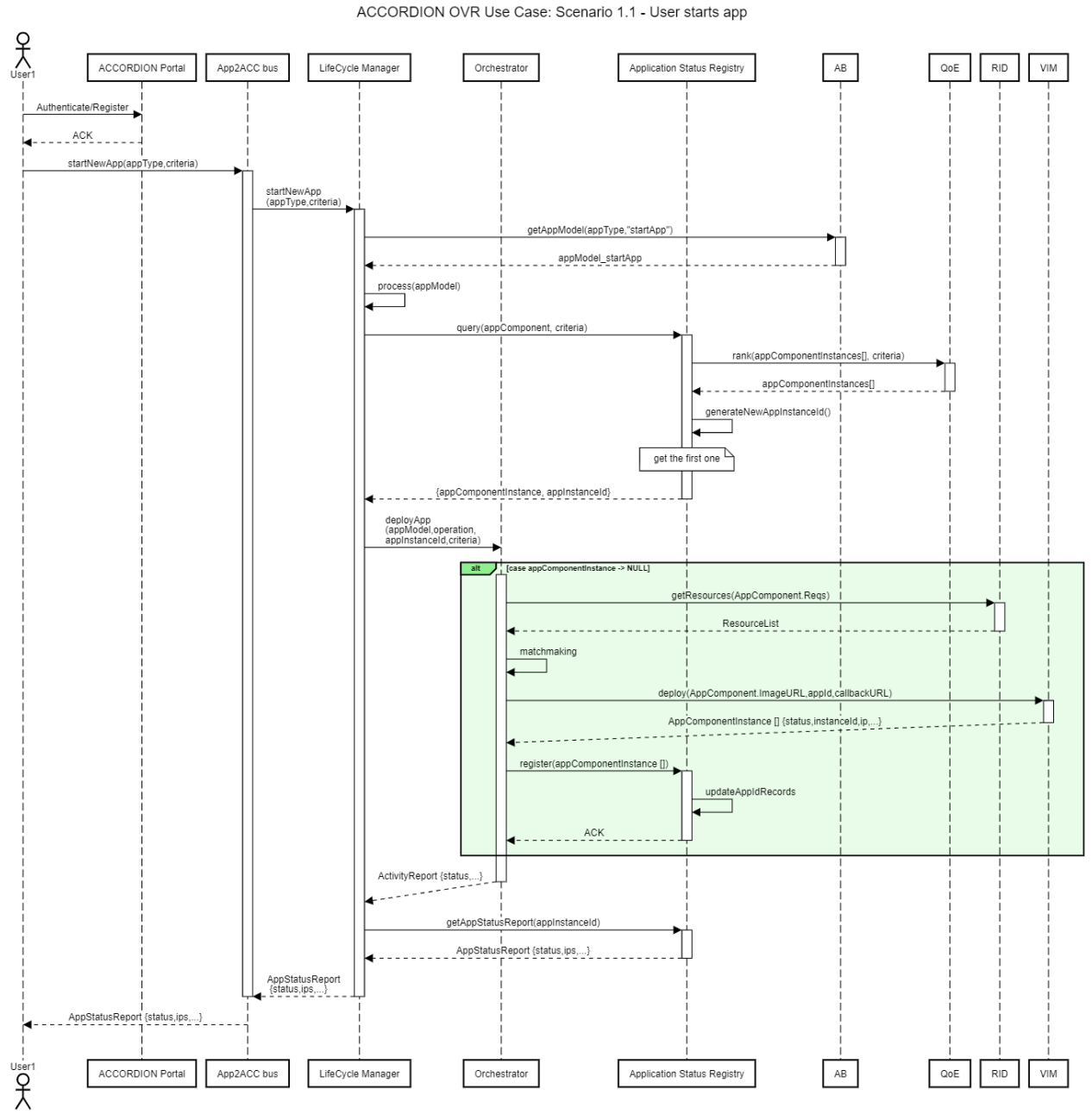

Figure 3-2: [OVR] Scenario 1.1 - User starts app

Scenario 1.1 User starts App: The user via the HMD is authenticated/registered in ACCORDION portal before invoking the startNewApp command that is published to the ACCORDION Bus (App2ACC). The parameter appType refers to application-instance-id described in sec. 3.4. The Lifecycle manager listens to the message and requests in a follow-up message the application model from the Application Bucket. Then it processes the data retrieved and queries from the Application Status Registry the state of the application and running instances. The QoE model supports a ranking mechanism based on the criteria provided to select the best candidate of the deployed application components and id. In case of an empty list the deployment process

described in sec. 3.5 follows, deploying the signaling server and the LS part, otherwise the Lifecycle Manager informs the Application Status Registry on the selected applicationInstanceID and the user via the App2ACC retrieves the Ips of the application components to further request direct communication.

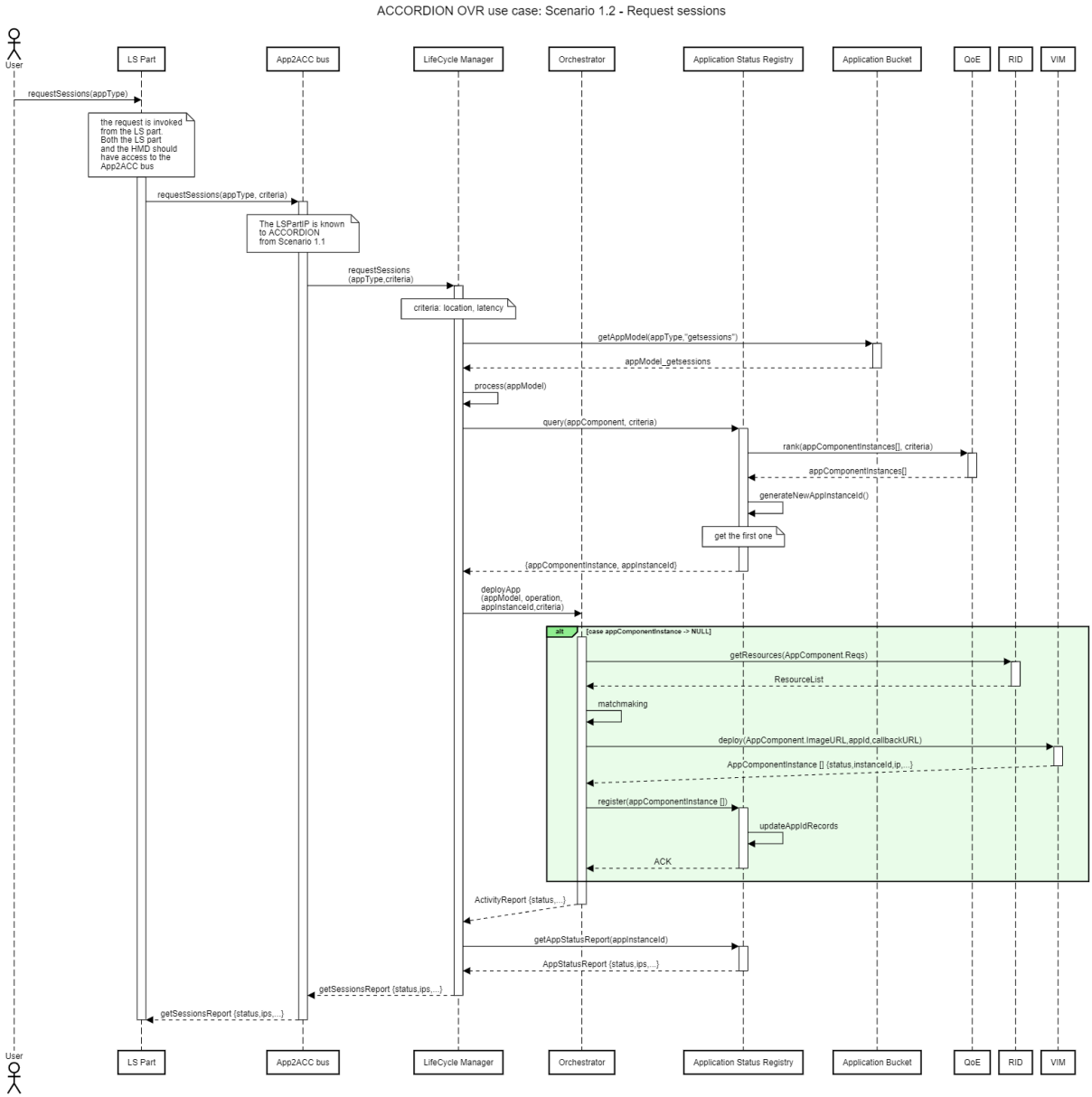

Figure 3-3: [OVR] Scenario 1.2 - Request sessions

Scenario 1.2 - Request session: In this scenario we consider that the user via the HMD has successfully established connection to the LS part through Scenario 1.1 and further wants to create or join a session in order to perform the VR training in collaborative mode. Although the UI input is provided by the HMD, the message for sessions request is sent to the App2ACC from the LS part, integrating the MAGES SDK

functionalities and the main processing of the application. Then the sequence of messages exchanges between the components of the platform follows Scenario 1.1 description.

3.6.3 Requirements

The following list provides an overall view on the status in addressing those requirements of UC1, as described in deliverable D2.1 and to be supported in Cycle I.

Table 3-2: Status of UC1 requirements

ID	Description	Req. level	Status
F_UC1_01	APPLICATION DEVELOPER: Use the mirror networking service or similar for matchmaking, creation of session and selection of an already existing session (ip, location, userid master)	Must	Due to the difficulties experienced with Mirror to support the relay server implementation the Photon networking is exploited.
F_UC1_02	USER: Able to create session and/or select an existing session from the application on the HMD based on my credentials	Must	Partly implemented
F_UC1_03	APPLICATION DEVELOPER: Session management through a relay server or message broker in the cloud	Must	The Photon networking is implemented. Alternative solutions were in parallel being investigated (MLAPI +Photon, Photon + Steam, DarkRift)
F_UC1_05	APPLICATION DEVELOPER: Communication of the application to Azure Cloud to store and retrieve user analytics	Must	Part of MAGES 3.0, supported by the application component running on the edge node
F_UC1_06	USER: Able to visualize my performance analytics on the HMD	Should	Part of MAGES 3.0, supported by the application component running on the edge node
F_UC1_07	APPLICATION DEVELOPER: The application component running on the HMD should be aware of the connected resources where part of the application has been offloaded	Must	As part of Scenario 1.1 the HMD retrieves the IP of the LS part via ACCORDION in order to establish direct communication.
F_UC1_08	APPLICATION DEVELOPER: The application running on the HMD should be able to connect via standardised protocols to the	Must	The communication to the LS part is done via UDP (WebRTC).

	resources where part of the application has been offloaded.		
F_UC1_09	APPLICATION DEVELOPER: Able to send output from the controllers and HMD to the resource where part of the application has been offloaded.	Must	Information is exchanged between the HMD and the LS part.
F_UC1_10	APPLICATION DEVELOPER: Able to receive encoded image data to the HMD for display from the resource where part of the application has been offloaded	Must	Information is exchanged between the HMD and the LS part.
F_UC1_11	APPLICATION DEVELOPER: Support continuous streaming of two images (one per eye) per user from the edge resource node to the HMD.	Must	Information is exchanged between the HMD and the LS part. A single texture is streamed to and cropped on the HMD.
F_UC1_12	APPLICATION DEVELOPER: Application component on the untethered HMD needs to be deployed and run-on ARM based architecture (to support SoC)	Must	The application component on the HMD is installed by the user or application developer.
F_UC1_16	USER, APPLICATION DEVELOPER: Able to start the application from the HMD without any additional system configuration.	Must	The deployment of the LS part instance is handled by the ACCORDION platform, The user (HMD) is registered /authenticated in the ACCORDION portal.
F_UC1_17	APPLICATION DEVELOPER: Communication of the HMD to the Remote Service (RS) in the cloud when launching the app on the HMD.	Must	The communication to the relay server is via the LS part and following successful communication of the HMD to the LS part.
F_UC1_18	APPLICATION DEVELOPER: Establish communication and initial data transfer from edge node for the user to enter the virtual OR.	Must	Communication and data transfer between the HMD and the LS part based on Scenario 1.1.

F_UC1_19	APPLICATION DEVELOPER: VR scene, data assets and avatars are locally stored on the edge node	Must	The LS part is deployed in the minicloud resources (edge node).

3.6.4 Prototype Implementation

<table>
<tr><th>Pilot #1</th><th>Collaborative VR module (CVRSB)</th></tr>
<tr><td></td><td></td></tr>
<tr><td>Experiment's Objectives</td><td>
<table>
<tr><td></td><td></td><td>Yes</td><td>No</td><td>Partly</td></tr>
<tr><td rowspan="3">Objective 4: Realize NextGen applications</td><td>Collaborative VR prototype</td><td></td><td></td><td>✓</td></tr>
<tr><td>Multiplayer mobile gaming</td><td></td><td></td><td></td></tr>
<tr><td>QoE optimization on content delivery</td><td></td><td></td><td></td></tr>
</table>
Objective 4 is partly covered since tests with 10 users will not be done in this cycle
</td></tr>
<tr><td>Experimental Workflow (Steps)</td><td>
1: Setup of Test Environment
The components of UC#1 are the signaling server, the LS part and the relay server
The main goals of UC#1 testing in Cycle I, are:
1. To verify that the LS part and the user's client application are able to establish communication
2. To test that the relay server is able to host multiple sessions for all the connected LS parts, with the core cooperative features being functional
2: Pre-validation testing
ACCORDION administrators need to make sure that:
1. The application components can be deployed and started properly
2. The signalling server's IP must be sent (or otherwise be identified) to the LS part and the client application
The two predefined ports of the relay server are bound, with the relay server also being made aware of its external IP address
3: Deploy application
In Cycle I, the application components can be deployed manually, but will require some manual operation in order for the appropriate communications to be set-up.
4: Start Application
</td></tr>
</table>

<table>
<tr><td></td><td colspan="3">When the signalling server and the LS part are deployed, the LS part will connect to the signalling server, as well as the client application.
Once both components of the computation offloading are connected to the signalling server, rendition streaming will start automatically.
5. User requests sessions
The user has to click on the “Online Session” button, and the networking interface should appear. They will then need to click on the button labelled: “Create Session".</td></tr>
<tr><td>Testbeds and Resources to be used</td><td colspan="3">Client application: Oculus Quest 2 for the client application
LS part: KVM enabled Linux machine with Nvidia GPU and VM
Relay Server: Linux machine
Signalling Server: Linux machine with Docker runtime installed</td></tr>
<tr><td>Metrics & Criteria to be measured</td><td colspan="3">The evaluation of UC#1 will address Objective 4 and more specifically it will aim to assess latency as part of KPI-O4-E1-1 and number of users as part of KPI-O4-E1-2
The impact KPIs to be addressed will be:
- iKPI-2: Improvement in user experience across each use case compared to the current state
- iKPI-5: Number of end-devices supporting interoperability and access via different VR/AR HMDs with few users (3-5 users)
The relevant requirements that will addressed in Cycle I are mentioned in sec. 3.6.3.
Details of the methods, tools & baseline values to be provided in D6.5</td></tr>
<tr><td>Experimental Subjects (participating on any of the steps above)</td><td>Role</td><td>Steps involved</td><td>No of participants</td></tr>
<tr><td></td><td>ACCORDION Administrator responsible for the deployment of the signaling server, the LS part and the relay server.

App Developer must provide the client application, as well as install the client application

End user responsible for launching the client App and connecting to the signaling server</td><td>ACCORDION Administrator: Deployment of the signalling server.

ACCORDION Administrator: Deployment of the LS part.

ACCORDION Administrator: Deployment of the relay server.

End-user launches client application and connects to the signalling server through its IP address.</td><td>The evaluation will be conducted with a small set of participants (3-5). Additional users may be recruited to support iKPI-2</td></tr>
</table>

3.7 Use Case#2: Multi-player Mobile Gaming

3.7.1 Pilot Prototype description

Use Case #2 provided by ORBK is a Mobile Multiplayer Game. The game system will consist of two main elements:

- Game Server

Mobile ApplicationGame Servers will be deployed at the Accordion infrastructure. Mobile Application will be run by the end-users – players, on their own mobile devices. The game system must be able to handle up to 100 players, handle a huge number of in-game events while performing full simulation of the game world and generate responses with minimal possible delay.

ORBK developed first version of the Game Server and prepared effective pipeline to create docker image of the server application. It is now ready for deployment testing.

First version of the Mobile Application is also ready for initial testing.

There is also the third element of the game system called: Game Servers Status DB that is responsible for gathering and handling all the information about Game Servers statuses along with some crucial information necessary for the game system correctly and to provide this information to the ACCORDION system.

Initial version of the Game Servers Status DB is also ready for testing. In Cycle I of the ACCORDION Project Game Servers Status DB will be deployed at AWS infrastructure outside the ACCORDION Federation. During Cycle II of the ACCORDION Project, we will consider running the Game Servers Status DB inside of the ACCORDION Federation.

3.7.2 Scenarios

In D6.1 we presented 3 scenarios:

- Submission of the Application – common for all Use Cases
- Start Application
- Runtime Adaptation

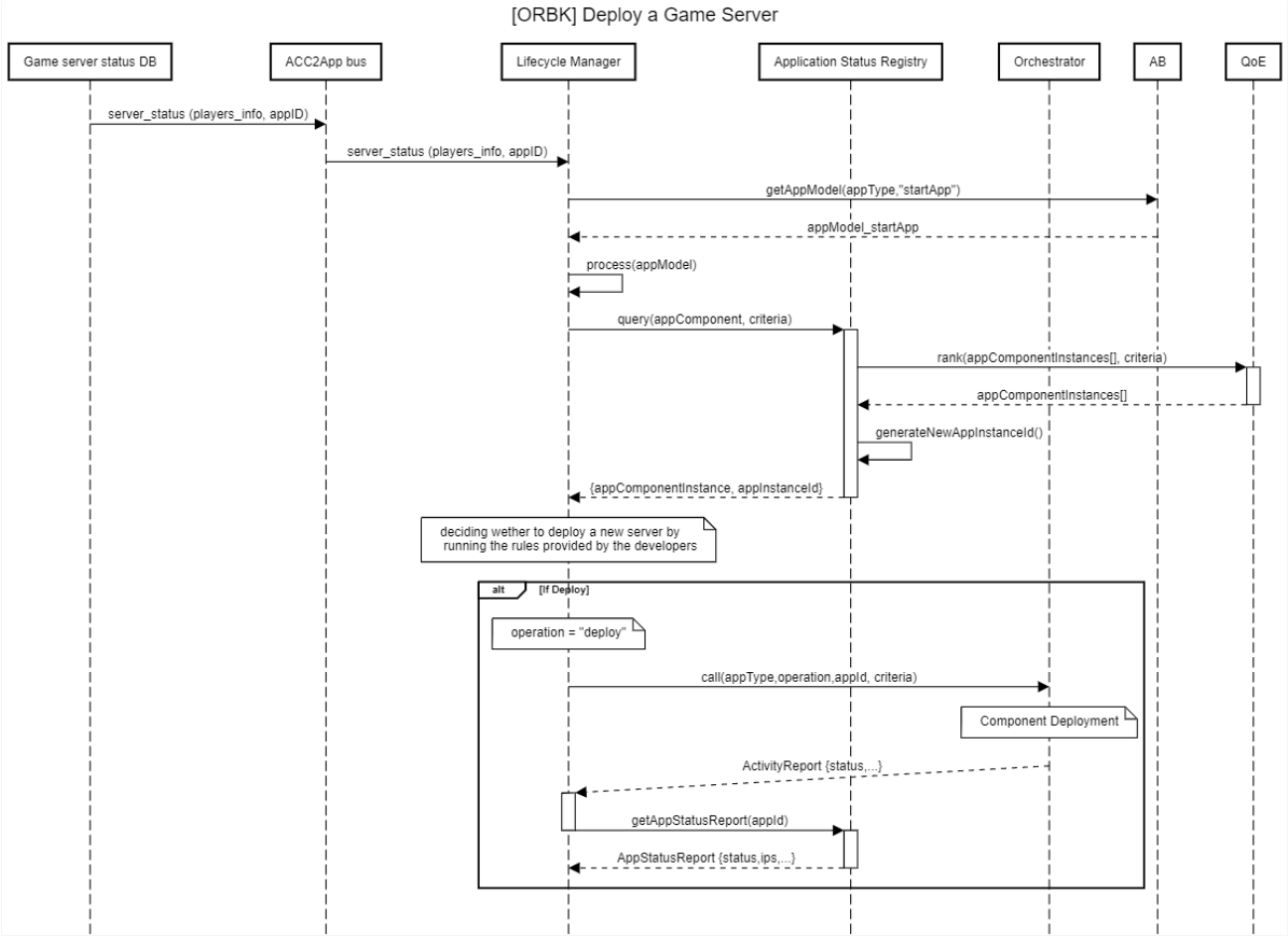

Figure 3-4: [ORBK]: Deploy a Game Server

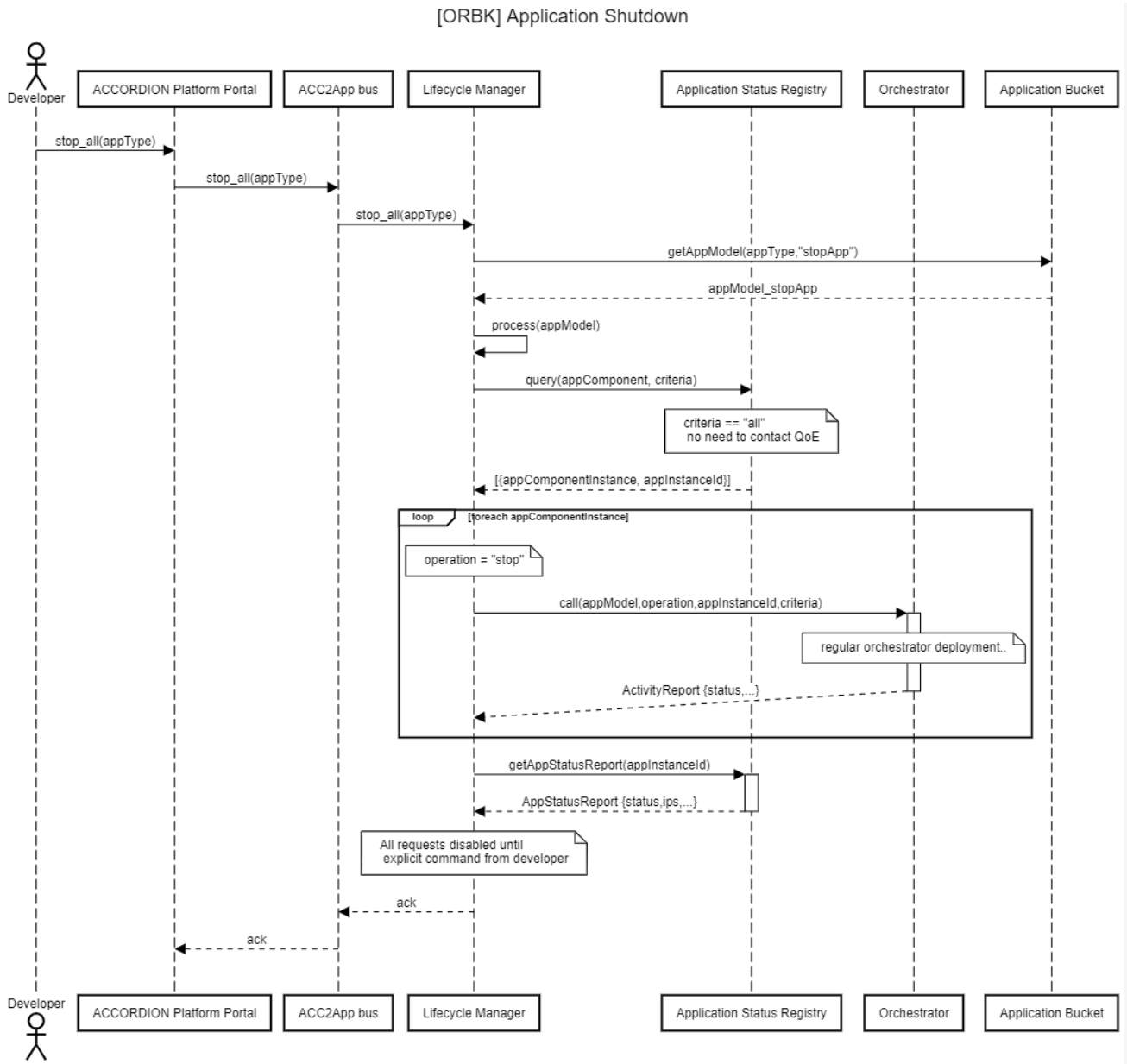

Figure 3-5: [ORBK] Application Shutdown

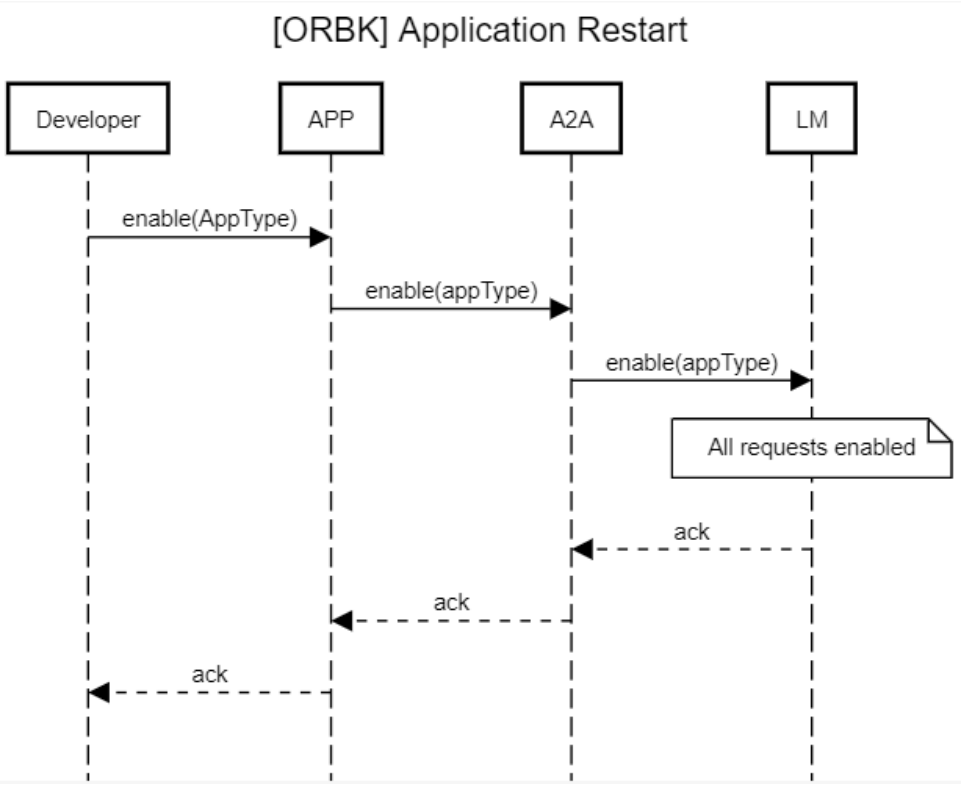

Figure 3-6: [ORBK] Application Restart

All 3 scenarios will be part of Cycle I of ACCORDION Project. In Cycle II we will focus our effort on optimising the game system in order to improve QoE of UC#2. We also plan to work on perfecting cooperation of the game system with the ACCORDION Platform.

3.7.3 Requirements

The following list provides an overall view on the status in addressing those requirements of UC2, as described in deliverable D2.1 and to be supported in Cycle I.

Table 3-3: Status of UC2 requirements

ID	Description	Req. level	Status
F_UC2_09	INFRASTRUCTURE: As ADMINISTRATOR I want to be able to see a list of all uploaded versions of the Game Server (Docker image file) with upload date and time and version number	Must	Ability to upload single Docker image of the Game Server. Full functionality is planned for II cycle

F_UC2_22	SERVICES: As APPLICATION DEVELOPER I want to have a service that deploys Game Server	Must	Ability to deploy single Docker image of Game Server. Full functionality is planned for II cycle
NF_UC2_06	INFRASTRUCTURE: As ADMINISTRATOR I want to the web-based administration the panel to enable very fast upload times of the Docker image files	Must	Ability to upload single Docker image of Game Server. Full functionality is planned for II cycle
F_UC2_22	SERVICES: As APPLICATION DEVELOPER I want to have a service that deploys Game Server	Must	Ability to deploy single Docker image of Game Server. Full functionality is planned for II cycle

3.7.4 Prototype Implementation

<table>
<tr><th>Pilot #2</th><th>Mobile multiplayer game</th></tr>
<tr><td></td><td></td></tr>
<tr><td>Experiment's Objectives</td><td>
<table>
<tr><td></td><td></td><td>Yes</td><td>No</td><td>Partly</td></tr>
<tr><td rowspan="3">Objective 4: Realize NextGen applications</td><td>Collaborative VR prototype</td><td></td><td></td><td></td></tr>
<tr><td>Multiplayer mobile gaming</td><td></td><td></td><td>✓</td></tr>
<tr><td>QoE optimization on content delivery</td><td></td><td></td><td></td></tr>
</table>
In Cycle I of ACCORDION project we focus on developing strong fundaments for further game system optimalisation thus Objective 4 is partly realized. In the next Cycle of the project, we will aim for full implementation of assumptions in KPIs: KPI-O4-E2-1: Multiplayer mobile gaming, average latency: ≤ 30ms and KPI-O4-E2-2: Multiplayer mobile gaming, number of concurrent users: ≥ 100.
</td></tr>
<tr><td>Experimental Workflow (Steps)</td><td>1: Setup of Test Environment
There are three main components UC#2: Game Server, mobile application and Server Status DB.
The main goals of the UC#2 testing in Cycle I are:
- to verify if the Game Server instance is deployed and running properly
- test connection and communication between all application components.
2: Pre-validation testing</td></tr>
</table>

	ACCORDION administrators need to make sure that the deployed Docker Image of the Game Server will be able to connect with both Server Status DB service and mobile application. Particular, predefined ports must be properly configured and open for communication. ***3: Deploy application*** In Cycle I it is sufficient to manually deploy Docker image of the Game Server. When deployed and run it will automatically register itself at the Server Status DB service. Server Status DB will be then able to provide list of Game Servers that are currently running. If the deployed test Game Server will be visible from the mobile application it will mean that the game system is operating properly. ***4: Start Application*** In UC#2 the application is started whenever at least one Game Server is deployed and running.		
Testbeds and Resources to be used	UC#2 requires at least one Game Server to be deployed at some VM available in ACCORDION Federation. Server Status DB will be deployed outside ACCORDION Federation in Cycle I. Mobile applications will be running at Android compatible mobile devices.		
Metrics & Criteria to be measured	*The evaluation of UC#2 will address Objective 4 and more specifically it will aim to assess latency as part of KPI-O4-E2-1 and number of users as part of KPI-O4-E2-2* *The impact KPIs to be addressed will be:* - iKPI-2: Improvement in user experience across each use case compared to the current state - iKPI-8: Number of different types of mobile devices to test the solution *The relevant requirements that will addressed in Cycle I are mentioned in sec. 3.7.3.:* F_UC2_09, F_UC2_22, NF_UC2_06, F_UC2_22, NF_UC2_10, NF_UC2_11, NF_UC2_13		
Experimental Subjects *(participating on any of the steps above)*	***Role***	***Steps involved***	***No of participants***
	ACCORDION administrator must be able to deploy Docker image of Game Server. App developer must deploy Server Status DB upload mobile application to the Play Store. End user must be able to download, install and run mobile application.	- deployment of the Game Server - deployment of the Server Status DB - publishing mobile app in Play Store	It is sufficient to test the app by 2 users in Cycle I.

3.8 Use Case#3: QoE optimization in content delivery

3.8.1 Pilot Prototype description

UC#3 consists of two sub use cases and one application is provided for each sub use case. The first application is a digital signage solution with proximity calculations and the second is an augmented reality loyalty mobile game. Both make use of the platform to provide ensure lower latency and higher performance on average network conditions.

The digital signage solution consists of 2 components:

- The application running on the edge (LS part)
- The Anblick Client (RS part)

The LS part will be deployed at the ACCORDION PLATFORM. The LS part runs in a Raspberry Pi with an external WiFi antenna which detects probe requests from mobile devices within a specific area. Data from probe requests is processed in the LS part and a content decision is made based on calculations performed on these data. The content decision is communicated through a content tag being shared via API REST with the Anblick Client.

The Anblick Client receives the content tag from the LS part, selects the corresponding content and streams this with the connected TV screens in the area.

The augmented reality game also consists of 2 components:

- The application running on the edge (LS part)
- Mobile application

The LS part is similar to the first sub use case and is deployed at the ACCORDION PLATFORM. In the LS part the coordinates of the location of the device are calculated. The location of the device of each user is shared with the Mobile Application.

The mobile android application will be run by the end-users - players, on their own mobile devices equipped with AR Core. Based on the location of the user the position of the 3D object of the game scenario is adapted.

3.8.2 Scenarios

As part of D6.1 seven scenarios were described. Three scenarios of the digital signage subUC were planned to be supported in Cycle I, whereas the remaining four would be part of Cycle II. Furthermore, scenario 3.1.3 has been added to specify the request of an Anblick endpoint to stream the right content from Edge to screen.

The following three scenarios are part of Cycle I:

- Scenario 0: Submission of the Application – common for all Use Cases
- Scenario 3.1.1: Start Edge service application
- Scenario 3.1.3: Request Anblick endpoint

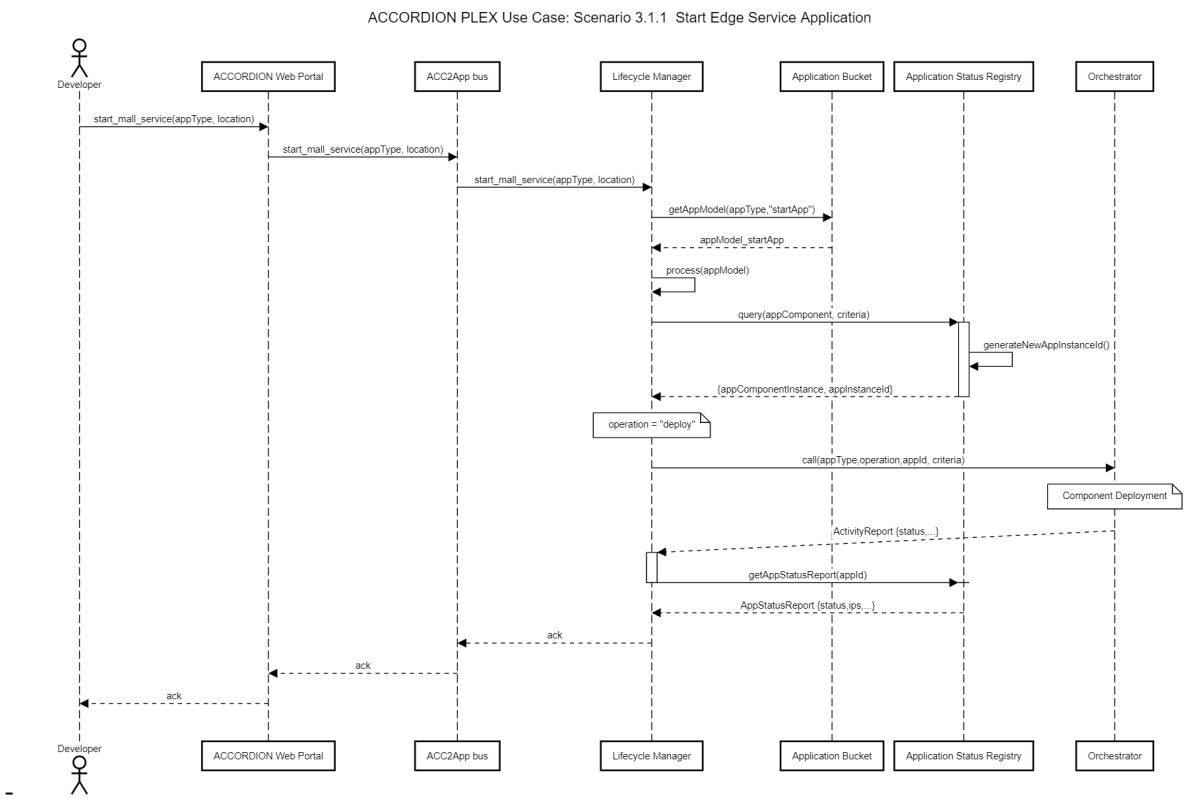

Figure 3-7: [PLEX] Start Edge Service Application

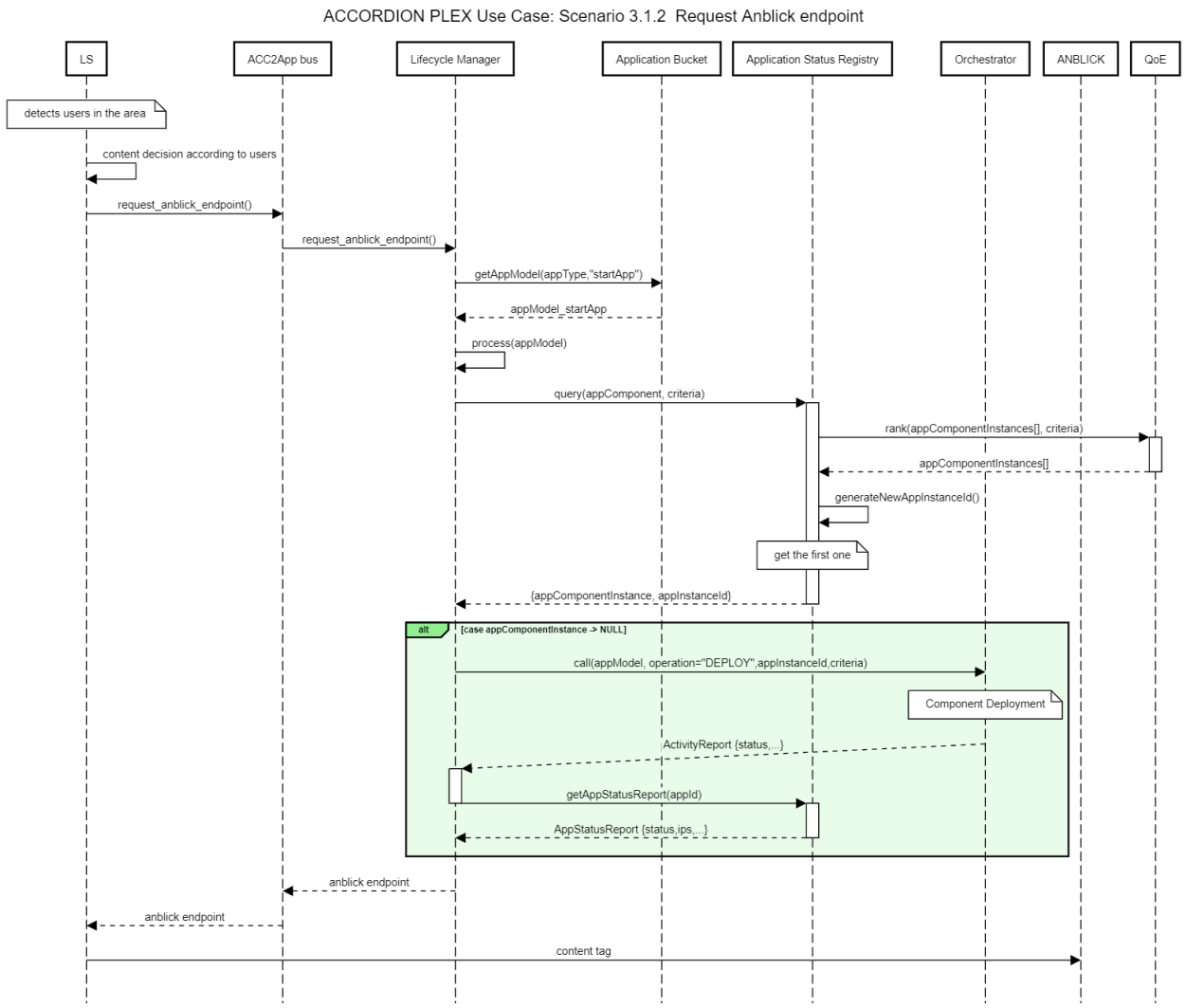

Figure 3-8: [PLEX] Request Anblick Endpoint

The augmented reality game subUC has been developed but has not been included in Cycle I to focus on the digital signage sub use case. In Cycle II we will further focus our effort on optimising the applications in order to improve QoE of UC#3. We also plan to work on the further integration of the use case with the ACCORDION Platform.

3.8.3 Requirements

The following list provides an overall view on the status in addressing those requirements of UC3, as described in deliverable D2.1 and to be supported in Cycle I.

Table 3-4: Status of UC3 Requirements

ID	Description	Req. level	Status

F_UC3_04	DEVICE IDENTIFICATION: As USER I want to Keep my privacy protected despite the connection so that I keep my privacy protected	Must	Non anonymized macs are doubly hashed for anonymization
F_UC3_05	DEVICE IDENTIFICATION: As ADMINISTRATOR I want to Check the number of connected devices and service status so that I can check the use of wifi	Must	Number of connected devices can be monitored as part of the Local System (LS)
F_UC3_12	DEVICE IDENTIFICATION: As DEVELOPER I want to get device data so that I can develop a platform	Must	LS receives device data
F_UC3_16	CLIENT IDENTIFICATION: As USER I want to Broadcasted information to be useful and personal related so that I stay better informed	Must	LS performs calculations on device characteristics and makes a content decision. Content tag is shared with RS.
F_UC3_17	CLIENT IDENTIFICATION: As USER I want to Data processed does not threaten my privacy so that I feel I am in a legal protected environment	Must	Double hashing and noise techniques
F_UC3_25	CLIENT IDENTIFICATION: As APPLICATION DEVELOPER I want to Get device data and client behavior info so that I can develop a better platform	Must	LS receives and device data
F_UC3_26	CONTENT DECISION MAKING: As USER I want to Decision making rules to be fair and honest so that content is not biased and becomes interesting and relevant	Must	LS performs calculations on device characteristics and makes a content decision based on device brand, position, volume, frequency.
F_UC3_29	CONTENT DECISION MAKING: As USER I want to Limit issues or items about my data for the decision making so that I retain control over the aspects of my data I want to be affected.	Must	Only data provided by probe request are processed
F_UC3_49	ADAPTED CONTENT: As USER I want to not publicly reveals my habits so that so	Must	Only anonymized data provided by probe request are processed.

	that no compromising information is shown		
F_UC3_58	CONTENT STREAMING: As DEVELOPER I want to keep transparent for other actors interactions with the cloud so that they are allowed to focus on other areas	Must	Transparent
F_UC3_59	CONTENT BROADCASTING: as an DEVELOPER I want to broadcast content in an instant manner to reduce traffic, avoid lags.	Must	RS (anblick client) streams content to screens
F_UC3_60	CONTENT STREAMING: As DEVELOPER I want to that Edges and Cloud are able to interchange info without human actions and solving possible problems so that Service integrity is kept	Must	LS and RS exchange data

3.8.4 Prototype Implementation

<table>
<tr><th>Pilot #3</th><th>QoE optimization on content delivery</th></tr>
<tr><td></td><td></td></tr>
<tr><td>Experiment's Objectives</td><td>
<table>
<tr><th></th><th></th><th>Yes</th><th>No</th><th>Partly</th></tr>
<tr><td rowspan="3">Objective 4: Realize NextGen applications</td><td>Collaborative VR (AR) prototype</td><td></td><td></td><td></td></tr>
<tr><td>Multiplayer mobile gaming</td><td></td><td></td><td></td></tr>
<tr><td>QoE optimization on content delivery</td><td></td><td></td><td>✓</td></tr>
</table>
QoE optimization on content delivery is the key objective of use case 3. In this first cycle content is adapted to the characteristics of the users but optimization of QoE is foreseen in cycle 2. Hence this objective is partly covered. The other objectives are related to the augmented reality loyalty game which has been developed but is not included in the prototype integration of Cycle I.
</td></tr>
</table>

<table>
<tr><td>Experimental Workflow (Steps)</td><td colspan="3">1: Setup of Test Environment
There are two main components of UC#3 for cycle I: The application running on the edge (LS part) , The Anblick Client (RS part)
The main goals of the UC#3 testing in Cycle I are:
- to verify if the Local System is deployed and running properly
- test connection and communication between all application components.
2: Pre-validation testing
ACCORDION administrators need to make sure that the deployed Docker Image of the LS part will be able to connect with the Anblick Client within the ACCORDION platform.
3: Deploy application
In Cycle I it is sufficient to manually deploy Docker image of the LS Part and Anblick Client. When deployed and run the LS part will automatically connect with the Anblick Client.
4: Start Application
In UC#3 the application is started whenever at least one Local System (LS) is deployed and running.</td></tr>
<tr><td>Testbeds and Resources to be used</td><td colspan="3">UC#3 requires at least one Local System (LS) and Anblick Client to be deployed at some VM available in ACCORDION Federation. To deploy the LS a Raspberry Pi with external wifi antenna is required as part of the testbed.</td></tr>
<tr><td>Metrics & Criteria to be measured</td><td colspan="3">The evaluation of UC#3 will address Objective 4 and more specifically it will aim to assess latency as part of KPI-O4-E3-1 and number of users as part of KPI-O4-E3-2
The impact KPIs to be addressed will be:
- iKPI-2: Improvement in user experience across each use case compared to the current state
- iKPI-11: Number of different types of mobile devices to test the solution
The relevant requirements that will addressed in Cycle I are mentioned in sec. 3.8.3. F_UC3_4, F_UC3_5, F_UC3_12, F_UC3_16, F_UC3_17, F_UC3_25, F_UC3_26, F_UC3_29, F_UC3_58, F_UC3_59, F_UC3_60
Details of the methods, tools & baseline values to be provided in D6.5</td></tr>
<tr><td rowspan="2">Experimental Subjects (participating on any of the steps above)</td><td>Role</td><td>Steps involved</td><td>No of participants</td></tr>
<tr><td>ACCORDION administrator must be able to deploy Docker image of LS and Anblick Client.
App developer must monitor interaction with Anblick,
End user must be able to see content on screen (digital signage)</td><td>- deployment of the LS and Anblick Client
- interaction with Anblick
- visualization of content</td><td>It is sufficient to test the use case by 2 users in Cycle I.</td></tr>
</table>

4 Use Cases Assessment Metrics

The forthcoming pilot execution and evaluation phase demonstrates that the traceability among use case KPIs, experiments' goals and experimental indicators is assured. To assist this alignment process, two feedback aspects will be taken into account:

• A *quantitative evaluation*, focusing on the evaluation of the technical quality of the ACCORDION platform in parts and as a whole using appropriate indicators and associated metrics. The main stakeholders who will be involved in this process are the technology providers.

• A *qualitative evaluation*, focusing on the experimental evaluation of the ACCORDION platform in real business settings and against user requirements (defined during the planning phase).

The aim of these assessments is to reach an agreed set of indicators against which validation of the 'success' will be measured. Such indicators may reflect technology features at component and platform level (e.g., operational performance), or business key performance indicators (e.g., customer satisfaction, latency levels, concurrent users etc.). The former set of metrics is use case independent, whilst the latter reflects the specific needs expressed in each use case. Alignment of both types of indicators is a main objective of the experimental definition phase. For each indicator, a set of quantifiable metrics is also defined, whose measurement relates to the achievement (or not), of a specific indicator. The execution of these evaluations is an ongoing process which also follows an iterative approach leading to several refinements performed during online workshop sessions as well as during online interviews with the use case providers. The results of this process will also form the baseline for the next ACCORDION prototype releases.

ACCORDION achievements are closely monitored through the Key Performance Indicators (KPIs) defined by the consortium, regarding the main ACCORDION objectives (section 4.1) and the overall impact KPIs (section 4.2) that ACCORDION envisions to achieve.

4.1 KPIs of ACCORDION Objectives

ACCORDION vision is realized through specific goals and main outcomes that depend on a number of intermediate challenges. In order to give to these challenges form, we have identified specific objectives and subsequent enablers, the achievement of which signifies the achievement of the ACCORDION mission.

The following table summarizes the KPIs of Objective 4 (Realize NextGen Applications) and subsequent enablers that will be monitored in the first evaluation cycle, based on the current implementation status of the ACCORDION platform.

Table 4-1: KPIs of ACCORDION Objective 4

Objective 4: Realize NextGen applications	To be monitored in First Cycle
KPI-O4-E1-1: Collaborative VR, average latency: ≤ 20ms	UC1
KPI-O4-E1-2: Collaborative VR, number of concurrent users: ≥ 15	UC1

KPI-O4-E2-1: Multiplayer mobile gaming, average latency: ≤ 30ms	UC2
KPI-O4-E2-2: Multiplayer mobile gaming, number of concurrent users: ≥ 100	UC2
KPI-O4-E3-1: Content delivery for cloud gaming applications, average latency: ≤ 20ms	UC3
KPI-O4-E3-2: Content delivery for cloud gaming applications, number of concurrent users: ≥ 20	UC3

4.2 Impact KPIs

ACCORDION, from the very beginning, has defined specific, ambitious objectives while increasing the impact in the research community by contributing to innovation capacity. ACCORDION contributions towards each of the expected impacts mentioned in the work programme are closely monitored through specific Key Performance Indicators (KPIs) and they are described in the following table (Table 4.2). The KPIs to be measured in the first evaluation cycle are reported below.

Table 4-2: Impact KPIs

Expected Impacts	KPI ID	ACCORDION Impact KPIs	Target	To be monitored in First Cycle
Contribute to the development of an ecosystem that will respond to the future digitisation needs of industry and the public sector	iKPI-1	Number of third-party application developers to support beta testing of the ACCORDION framework	**>5**	
Assist the development of new cloud-based services and infrastructures in Europe and foster an industrial capability in the cloud computing sector	iKPI-2	Improvement in user experience across each use case compared to the current state	**>20%**	UC1, UC2, UC3
Create new opportunities to encourage European-based providers, in particular SMEs, to develop and offer cloudbased services based on the most advanced technologies	iKPI-3	Validate better performance compared to existing off-the-shelf solutions in terms of latency, CCU's	**>15%**	
	iKPI-4	Number of initiatives, clusters to disseminate key project findings	**>5**	
VR/AR technologies	iKPI-5	Number of end-devices supporting interoperability and access via different VR/AR HMDs	**>10**	UC1
	iKPI-6	Increasing the average QoE by the end-users at each validation cycle	**>20%**	
	iKPI-7	Validating acceptance by end-users of the fully decentralized configuration of testbed	**>80%**	
Gaming Industry	iKPI-8	Number of different types of mobile devices to test the solution	**>5**	UC2

	iKPI-9	Increasing the average QoE by the end-users at each validation cycle	**>20%**	
	iKPI-10	Validating acceptance by end-users of the fully decentralized configuration of testbed	**>80%**	
Content Delivery	*iKPI-11*	*Number of different types of mobile devices to test the solution*	*>5*	*UC3*
	iKPI-12	*Increasing the average QoE by the end-users at each validation cycle*	*>20%*	
	iKPI-13	*Validating acceptance by end-users of the fully decentralized configuration of testbed*	*>80%*	

5 Conclusions

This deliverable reports on the progress of the specification of the operational experiments and facilitates the experimentation phase of the ACCORDION project. Special effort has been made to detail all experiment scenarios, according to the ACCORDION implementation plan, in terms of the environment preparation; experimental workflow specification and participants' identification. Also, it includes the definition of the experimental evaluation indicators according to the business objectives.

These experiments will be performed using the first release of the ACCORDION framework. The initial results will be reported in the first evaluation report (D6.5 due to M19). The specification and refinement of the operational experiments will continue during the experimentation phase and will drive the released functionalities of the ACCORDION platform (second implementation cycle).