



---

---

## **Deliverable D4.2**

### **Intermediate DECIDE ADAPT Architecture**

---

---

<b>Editor(s):</b>	Lorenzo Blasi
<b>Responsible Partner:</b>	Hewlett Packard Italiana / HPE
<b>Status-Version:</b>	Final - v1.1
<b>Date:</b>	26/11/2018
<b>Distribution level (CO, PU):</b>	PU

<b>Project Number:</b>	GA 731533
<b>Project Title:</b>	DECIDE

<b>Title of Deliverable:</b>	Intermediate DECIDE ADAPT Architecture
<b>Due Date of Delivery to the EC:</b>	30/11/2018

<b>Workpackage responsible for the Deliverable:</b>	WP4 - Continuous deployment and operation
<b>Editor(s):</b>	Hewlett Packard Italiana S.r.l. (HPE)
<b>Contributor(s):</b>	Lorenzo Blasi, Paolo Barone (HPE); Juncal Alonso (Tecnalia); Javier Gavilanes (Experis)
<b>Reviewer(s):</b>	Simon Dutkowski (FhG)
<b>Approved by:</b>	All Partners
<b>Recommended/mandatory readers:</b>	WP2, WP3, WP5, WP6

<b>Abstract:</b>	This deliverable describes the intermediate version of the architecture of DECIDE ADAPT tool providing a comprehensive overview of the system using different architectural views to represent different aspect of the system (e.g. Use Case View, Logical View, Process View, Deployment View).
<b>Keyword List:</b>	Adaptation, monitoring, deployment, scalability, private cloud, hybrid cloud
<b>Licensing information:</b>	This work is licensed under Creative Commons Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0) <a href="http://creativecommons.org/licenses/by-sa/3.0/">http://creativecommons.org/licenses/by-sa/3.0/</a>
<b>Disclaimer</b>	This document reflects only the author's views and the Commission is not responsible for any use that may be made of the information contained therein

## Document Description

### Document Revision History

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
v0.1	20/07/2018	Defined document ToC	HPE
V0.2	20/7/2018	ToC updated during WP4 call	HPE
v0.3	20/9/2018	Initial TECNALIA contribution	TECNALIA
V0.4	10/10/2018	Integrated initial HPE contribution (sect. 2.1 and 3.1)	HPE
V0.5	18/10/2018	Completed Architecture section, added initial Interfaces section	HPE
V0.6	22/10/2018	Completed DO Architecture, Hybrid scenario and Introduction sections.	HPE
V0.7	23/10/2018	Integrated first contribution from Experis and added Exec Summary and Conclusions	HPE, Experis
V0.8	24/10/2018	Updated missing references and updated section 3.2	HPE
V0.9	25/10/2018	Version almost ready for internal review	HPE
V0.10	31/10/2018	Added further Experis contribution in section 6.1	HPE, Experis
V0.11	16/11/2018	Updates after the internal review	TECNALIA, HPE
V0.12	19/11/2018	Updated MM internal components' Names and finalised editing	HPE
V1.0	19/11/2018	Final editing before release	HPE
V1.1	26/11/2018	Added further hybrid scenario	HPE

---

## Table of Contents

---

Table of Contents .....	4
List of Figures.....	5
List of Tables.....	5
Terms and abbreviations.....	6
Executive Summary .....	7
1 Introduction.....	9
1.1 About this deliverable .....	9
1.2 Innovation .....	9
1.3 Document structure .....	9
2 ADAPT requirements.....	10
2.1 Requirements planned for Y2.....	10
2.2 Requirements from Business Use Cases .....	13
3 High Level ADAPT Functionalities.....	16
3.1 Updated ADAPT UML Use Cases .....	16
3.1.1 ADAPT DO Use Cases .....	16
3.1.2 ADAPT MM Use Cases .....	19
3.1.3 ADAPT VH Use Cases .....	20
3.2 Monitoring scalability.....	21
4 DECIDE ADAPT Updated Architecture.....	24
4.1 ADAPT DO architecture .....	25
4.2 ADAPT MM architecture .....	26
4.3 ADAPT VH architecture .....	29
5 DECIDE ADAPT Interfaces.....	31
5.1 ADAPT integration .....	31
5.2 Application Description .....	32
6 ADAPT deployment scenarios .....	34
6.1 ADAPT integration platform .....	34
6.2 ADAPT Hybrid scenario.....	35
6.3 ADAPT as a Service .....	36
7 Conclusions.....	38
8 References.....	39

---

## List of Figures

---

<b>FIGURE 1.</b> ADAPT DEPLOYMENT ORCHESTRATOR USE CASES DIAGRAM.....	16
<b>FIGURE 2.</b> DECIDE ADAPT MM USE CASES DIAGRAM.....	19
<b>FIGURE 3.</b> VIOLATIONS HANDLER’S USE CASES .....	21
<b>FIGURE 4.</b> DATABASE MANAGEMENT SYSTEMS RANKING ACCORDING TO POPULARITY (SOURCE DB-ENGINE [11]) .....	22
<b>FIGURE 5.</b> TOP 10 TIME SERIES DATABASES AND THEIR HISTORICAL CHANGES (SOURCE DB-ENGINE [12]) .....	23
<b>FIGURE 6.</b> DECIDE ADAPT LOGICAL ARCHITECTURE AND EXTERNAL INTERFACES .....	24
<b>FIGURE 7.</b> ADAPT DEPLOYMENT ORCHESTRATOR COMPONENTS .....	26
<b>FIGURE 8.</b> ADAPT MONITORING MANAGER (MM) INTERNAL COMPONENT DIAGRAM .....	27
<b>FIGURE 9.</b> EXCERPTS OF ADAPT MM UI .....	28
<b>FIGURE 10.</b> ADAPT MONITORING MANAGER EXTERNAL COMPONENT DIAGRAM .....	29
<b>FIGURE 11.</b> ADAPT VIOLATIONS HANDLER COMPONENT DIAGRAM .....	29
<b>FIGURE 12.</b> ADAPT VIOLATIONS HANDLER EXTERNAL INTERFACES DIAGRAM .....	30
<b>FIGURE 13.</b> INTERNAL AND EXTERNAL ADAPT INTERFACES AND THEIR STATUS.....	31
<b>FIGURE 14.</b> ADAPT INTEGRATION PLATFORM SCENARIO .....	34
<b>FIGURE 15.</b> ADAPT HYBRID SCENARIO .....	35
<b>FIGURE 16.</b> FURTHER ADAPT HYBRID DEPLOYMENT SCENARIO .....	36
<b>FIGURE 17.</b> EXAMPLES OF ARCHITECTURAL AND DESIGN QUESTIONS IN SAAS (SOURCE SEI [15]) .....	36

---

## List of Tables

---

<b>TABLE 1.</b> YEAR 1 REQUIREMENTS CLASSIFICATION .....	10
<b>TABLE 2.</b> MERGED REQUIREMENTS .....	11
<b>TABLE 3.</b> REQUIREMENTS FROM BUSINESS USE CASES .....	14
<b>TABLE 4.</b> CURRENT STATUS OF ADAPT INTERFACES.....	32

---

## Terms and abbreviations

---

AD	Application Description
CSP	Cloud Service Provider
DB	Database
DO	Deployment Orchestrator
EC	European Commission
MCSLA	Multi-Cloud SLA
MM	Monitoring Manager
NFR	Non-Functional Requirement
REST	Representational State Transfer
SLA	Service Level Agreement
SLO	Service Level Objective
ToC	Table of Contents
TSDB	Time Series Data Base
UI	User Interface
UML	Unified Modeling Language
VH	Violations Handler
VM	Virtual Machine
VPN	Virtual Private Network
WP	Work Package

## Executive Summary

This deliverable D4.2 (*Intermediate DECIDE ADAPT Architecture*) describes the current version of DECIDE ADAPT's high-level architecture. The architecture is described using standard UML syntax both for the Use Cases diagrams and the component and interfaces diagrams. We tried to keep the document short by including only the most relevant functional information. Other WP4 deliverables, D4.5 (*Intermediate multi-cloud application deployment and adaptation*) [1] and D4.8 (*Intermediate multi-cloud application monitoring*) [2] will describe in more detail the implemented components. This document also summarizes the current integration state and describes multiple deployment scenarios for ADAPT itself. This document is not intended as a user manual for ADAPT operation.

The architecture and components' functionalities are based on the requirements elicited. The first version of this document, D4.1 [3] listed all the collected WP4 requirements and described the related analysis, but a more in-depth analysis of the requirements for ADAPT discovered several duplicates. Therefore, in this second year we performed further requirements analysis work, both to improve the requirements classification and to reduce their number by merging requirements concerning the same functionality. This work, which reduced the number of requirements from 51 to 17, is reported in the first section after the introduction, section 2, and its result is summarized in a table which includes for each merged requirement, for traceability purposes, a mapping with the corresponding requirements identified in the first year. The concerned components and planned implementation deadlines are also included.

The same section 2 also lists the requirements from DECIDE's Business Use Cases that apply to ADAPT (from deliverable D6.2 [4]). The brief impact analysis reported for each requirement shows that most of them can be satisfied by the current Year 2 ADAPT release. Only two of these requirements will need further analysis and ADAPT extensions to be satisfied in the next ADAPT release: a list of metrics on redeployment and the support for containers replication.

The following section 3 includes updated UML Use Cases describing the functionalities of each of the three ADAPT components. The same section also includes a discussion about scalability for the monitoring functionality, which is an issue linked to the big amount of data it produces. Time-series data, such as monitoring metrics, accumulates very quickly and normal databases are not suitable for handling that scale, whereas Time Series Databases (TSDB) are built for handling massive amounts of time-stamped measurements and for measuring change over time. This section explains why DECIDE selected InfluxDB as the Time Series Data Base for ADAPT Monitoring Manager (MM).

Section 4 describes ADAPT architecture, initially at a high level and as well as all the external interfaces; the same section then describes in some more detail the internal structure of each ADAPT component. The main changes in the ADAPT architecture with respect to year 1 are: the integration with DECIDE DevOps Framework for displaying monitoring metrics, the new interface with the Application Controller to obtain the correct aggregation formulas for each SLA, and the updated interfaces with ACSml that now receive only violation information and start / stop CSP monitoring in sync with application's deployment.

Section 5 describes the integration status of both internal and external interfaces and summarizes the Application Description fields read or written by ADAPT. For the interfaces which have been recently redesigned the integration status is less advanced, whereas in most other cases it is progressing well. The Application Description document is the most important ADAPT interface and the main way of exchanging complex information between DECIDE tools. Section 5 summarizes the current status of information exchanged by ADAPT through the Application Description, indicating for each group of fields, which tool is the producer, and which is the consumer of the related information.

Finally, section 6 describes ADAPT deployment scenarios. The first scenario described has been already implemented for both integration and demo purposes, as a centralized deployment in the integration environment. A different deployment option, involving one ADAPT instance for each application and deployed with it, has been explored in this second year and then abandoned for two main reasons: to avoid moving historical monitoring data around on each redeployment, and to avoid developers to pay for additional resources for each application. The current idea is to have a central deployment for ADAPT which will handle multiple users and multiple applications for each user. Further requirements to accommodate such a multi-tenant deployment, such as scalability and security, will be explored in the next year. Section 6 also describes two additional deployment options: a hybrid scenario which is currently being implemented, and a possible future scenario to offer ADAPT-as-a-Service in the Cloud. Hybrid scenarios aim at supporting private clouds as well as public ones. ADAPT could support different hybrid scenarios, mainly distinguished by the location where ADAPT is executed, whether locally or in the cloud, and by the agent used for provisioning in the private cloud, either ACSml or ADAPT itself.

The main innovations reported in this second-year architecture deliverable are related or derived from the integration work. For example, the improvement of the interface between the ADAPT Monitoring Manager and ACSml: instead of collecting all the monitoring data about the cloud providers used by the application, now ADAPT only receives the information about SLA violations of those providers. Another innovation which is currently being implemented in the integration between ADAPT and ACSml is the support for hybrid scenarios, which can combine the benefits of public clouds with those of private ones. The integration work also led to the discovery of issues not foreseen in the first year, such as the need for a DECIDE component dedicated to keeping the various “secrets” involved in the deployment of an application; this is currently under analysis and it will be better detailed in deliverable D2.7.

In the next year the consortium will introduce a further hybrid scenario, with direct OpenStack provisioning, and will analyse the possibility to introduce additional functionalities, such as: new monitoring metrics on redeployment, container replication, scalability and security for multi-tenant deployments, and possibly also container volumes.



# 1 Introduction

## 1.1 About this deliverable

This deliverable is produced as result of the Task 4.1, *DECIDE ADAPT Architecture and Integration of DECIDE WP4 (Continuous deployment and operation)*. It describes the intermediate version of DECIDE ADAPT's high-level architecture. The architecture is described using standard UML syntax, both for the use cases and the component and interfaces diagrams. This deliverable includes the most relevant functional information about ADAPT. Other deliverables, D4.5 (*Intermediate multi-cloud application deployment and adaptation*) [1] and D4.8 (*Intermediate multi-cloud application monitoring*) [2] will describe in more detail the implemented components. This document also summarizes the current integration state and indicates multiple deployment scenarios for ADAPT itself. This document is not intended as a user manual for ADAPT operation.

## 1.2 Innovation

The research reported in this deliverable represents the logical continuation and improvement of the research reported in the first ADAPT Architecture deliverable, D4.1 [3]. The main topic tackled in this second year has been the integration of the components, both among them and with other DECIDE tools. This work resulted in a more detailed definition of the interfaces and the discovery of issues not foreseen in the first year, such as the need for a DECIDE component dedicated to keeping the various "secrets" that are needed for working in a secure integration of the various DECIDE components; more about this will be reported in deliverable D2.7. Another improvement derived from the integration work has been the modification of the interface between the ADAPT Monitoring Manager and ACSml: instead of collecting from ACSml all the monitoring data about the cloud providers used by the application, now ADAPT only receives the information about SLA violations of those providers, thus avoiding transmission and recording of a lot of data. Finally the integration work among ADAPT Deployment Orchestrator and ACSml Contracting is currently being extended to experiment the implementation of a hybrid scenario allowing multi-cloud deployments to also include private clouds.

## 1.3 Document structure

The first section after this introduction, section 2, reports about further requirements analysis work performed in the second year, both to improve the requirements classification and to reduce duplications by merging requirements concerning the same functionality. The following section 3 includes updated UML Use Cases describing the functionalities of each of the three ADAPT components. The same section also includes a discussion about scalability for the monitoring functionality, which is an issue linked to the big amount of data it produces. Section 4 describes ADAPT architecture, initially at an high level and also describing all the external interfaces; the same section then describes into some more detail the internal structure of each ADAPT component. The internal interfaces are then described in section 5 along with the integration status of both internal and external interfaces, and a summary of the Application Description fields read or written by ADAPT; the whole Application Description is described in full detail in deliverable D2.5 [5]. Finally section 6 describes ADAPT deployment scenarios, reporting about the current deployment in the integration / demonstration environment, the hybrid scenario which is currently being implemented, and a possible scenario to offer ADAPT-as-a-Service on the Cloud.

## 2 ADAPT requirements

### 2.1 Requirements planned for Y2

In the first year a requirement analysis work had already been performed, to identify two categories of requirements: those which apply to the whole DECIDE framework and not to ADAPT in particular (termed “General DECIDE requirements” in D4.1, section 2.3 [3]), and those which provide a useful high level description of the DECIDE ADAPT component (indicated as “ADAPT description” in D4.1, section 2.3). The remaining requirements have then been mapped on the identified ADAPT functional use cases (in D4.1 section 3.3).

The requirements’ classification, in addition to the General requirements and ADAPT description, now marks also requirements which are not about ADAPT, i.e. concerning other DECIDE components, and requirements which are not functional but dictate a specific architecture. Both classes of requirements have been discarded, along with many requirements marked ADAPT Description and almost all those marked General DECIDE requirement. Also the KPI definitions have not been included in the final list since KPIs will be evaluated in deliverables D6.5 and D6.6. The classification of the full list of WP4 requirements is shown in

**Table 1.** The last three columns of the table indicate which of the three ADAPT components, i.e. Deployment Orchestrator (DO) Monitoring Manager (MM) and Violations Handler (VH), are impacted.

**Table 1.** Year 1 requirements classification

Id	General DECIDE req.	ADAPT description	ADAPT architecture	KPI	Not about ADAPT	DO	MM	VH
WP4-REQ1						X	X	X
WP4-REQ2							X	
WP4-REQ3						X	X	X
WP4-REQ4							X	
WP4-REQ5				X				
WP4-REQ6				X				
WP4-REQ7	X							
WP4-REQ8		X				X	X	X
WP4-REQ9		X				X	X	
WP4-REQ10		X						
WP4-REQ11	X							
WP4-REQ12	X				X	X		
WP4-REQ13					X			
WP4-REQ14						X		
WP4-REQ15					X	X		
WP4-REQ16					X			
WP4-REQ17	X							
WP4-REQ18							X	
WP4-REQ19					X			
WP4-REQ20					X			
WP4-REQ21						X		X
WP4-REQ22								X
WP4-REQ23						X		
WP4-REQ24					X			
WP4-REQ25	X							

Id	General DECIDE req.	ADAPT descrip- tion	ADAPT architec- ture	KPI	Not about ADAPT	DO	MM	VH
WP4-REQ26					X			
WP4-REQ27							X	X
WP4-REQ28			X			X		
WP4-REQ29			X				X	
WP4-REQ30			X			X		
WP4-REQ31			X					
WP4-REQ32			X					
WP4-REQ33			X					
WP4-REQ34						X	X	
WP4-REQ35		X				X		
WP4-REQ36		X				X		
WP4-REQ37		X						
WP4-REQ38				X				
WP4-REQ39				X				
WP4-REQ40				X				X
WP4-REQ41	X	X					X	
WP4-REQ42	X							
WP4-REQ43								X
WP4-REQ44		X				X		
WP4-REQ45		X						
DEVOPS-REQF3							X	
DEVOPS-REQF6						X		
DEVOPS-REQF9	X						X	
DEVOPS-REQF15							X	
DEVOPS-REQF16						X		
DEVOPS-REQF17	X				X		X	

The resulting set of ADAPT requirements was redundant, therefore a further analysis work has been done to reduce their number by merging requirements concerning the same functionality. The result of this work is shown in the **Table 2** below and includes, for traceability purposes, a mapping with the requirements identified in the first year. The concerned components and planned implementation deadline are also included.

**Table 2.** Merged requirements

Req. ID	Y1 Req ID(s)	Description	Component	Deadline
WP4-MR1	WP4-REQ1, WP4-REQ3, WP4-REQ8, WP4-REQ10, DEVOPS-REQF16	DECIDE ADAPT will support the semi-automatic adaptation and dynamic re-deployment of (parts of) multi-cloud applications when certain conditions are not met, by changing the configuration and topology of services at operational time based on continuous monitoring of both the conditions of the application and the CSPs where the application is deployed on..	DO, MM, VH	M24

Req. ID	Y1 Req ID(s)	Description	Component	Deadline
WP4-MR2	WP4-REQ2	A violation is raised when the defined composite multi-cloud application SLA is not being fulfilled, the application is not performing as established or the cloud service providers (CSPs) are violating the contracted SLAs.	MM	M24
WP4-MR3	WP4-REQ4, WP4-REQ18, DEVOPS-REQF3, DEVOPS-REQF15, DEVOPS-REQF17	ADAPT will monitor the objectives indicated in the MCSLA, monitoring the metrics related to the application components (micro-services) and gathering the run-time information related to the CSPs monitoring from other components (ACSMI).	MM	M12
WP4-MR4	WP4-REQ9	DECIDE ADAPT will support automated dynamic deployment of service components	DO	M12
WP4-MR5	WP4-REQ12	DECIDE ADAPT will generate scripts for automating both resource provisioning and deployment for multi-cloud native applications	DO	M12
WP4-MR6	WP4-REQ14, WP4-REQ23	ADAPT will support manual checking of the deployment configuration and scripts	DO	M30
WP4-MR7	WP4-REQ15	ADAPT will maintain the current deployment configuration situation; other tools will maintain the history of the previous deployment configurations, so that they can be checked in the re-deployment phase	DO	M12
WP4-MR8	WP4-REQ21	In case the technological risk for the application has been defined as low, the multi-cloud application will be redeployed automatically, following a new deployment configuration [provided by triggering OPTIMUS].	VH, DO	M24
WP4-MR9	WP4-REQ22, WP4-REQ43	In case the technological risk for the application has been defined as high, once ADAPT has identified the violation(s) that are affecting the malfunctioning of the application, it will both alert the operator and trigger OPTIMUS, sending it the identified violation(s), to simulate a new deployment configuration that could avoid the same problem.	VH	M30

Req. ID	Y1 Req ID(s)	Description	Component	Deadline
WP4-MR10	WP4-REQ27	In case of a violation, ADAPT will report to the operator the NFR (SLO) that are not being fulfilled	MM, VH	M24
WP4-MR11	WP4-REQ34, WP4-REQ37	ADAPT functionalities (deployment, monitoring and adaptation) rely on information about the multi-cloud application obtained from the Application Description	DO, MM, VH	M12 - M24
WP4-MR12	WP4-REQ35	ADAPT will support applications based on composition of stateless (possibly micro) services	DO	M12
WP4-MR13	WP4-REQ36	ADAPT will support composable applications where each composition unit is a containerized service	DO	M12
WP4-MR14	WP4-REQ41	DECIDE (and ADAPT in particular) will support extensions to add more NFRs that need to be measured.	MM	M30
WP4-MR15	WP4-REQ44	Users will perceive relevant improvements in the business continuity since as soon as there is a problem (i.e. lack of resource due to a peak of requests) the software is automatically re-adapted and re-deployed	DO	M24
WP4-MR16	DEVOPS-REQF6	DECIDE ADAPT will support the continuous deployment of the developed apps	DO	M24
WP4-MR17	WP4-REQ15	DECIDE will maintain and store a history of the violations occurred for a deployed application	MM	M24

## 2.2 Requirements from Business Use Cases

This section lists the requirements from use cases (see deliverable D6.2 [4]) that apply to ADAPT and it provides a brief analysis of the impact on ADAPT for each of them. Impacts to be analyzed for the next release are highlighted (underlined).

**Table 3.** Requirements from Business Use Cases

Req. ID and Title	Description	Impact on ADAPT
AMR03 Management of Deployed Cloud Environments	Management of Cloud infrastructure, including networks, through dashboard(s)	Infrastructure management is out of scope for ADAPT, and products for this are available on the market <sup>11</sup> , but at least the status of the current deployment can be viewed through the ADAPT section in the DevOps Framework
AMR04 Deployment of Software from repository	Ability to designate software repository and have DECIDE deploy it directly from that repository	Available in the current release: a private registry for containers can be specified in the Application Description
ARR06 Monitoring and Re-deployment Services	OPTIMUS will support the provision of new possible topologies when a violation occurs, triggering the redeployment process. OPTIMUS will consider past deployment configurations in order to develop and to propose the new ones. ACSmI will be able to monitor the CSPs (SLAs) and in case of a violation of the SLA, will inform to ADAPT. ADAPT will confirm if a new redeployment is required.	Application redeployment is planned for this M24 release: violations will be collected by ADAPT Violations Handler, which will trigger a new redeployment by asking OPTIMUS for a new deployment schema
ARR08 Objective Measures of Performance in Multi-cloud	ACSmI monitoring will define the metric/parameters to be monitored for each NFR. ACSmI monitoring will monitor these parameters in the CSPs and will send an alert to ADAPT (violation handler) in case any of the measured metrics do not fulfil the agreed SLA. The metrics will be defined in the next months of the project. Minimum measures to be provided: Average availability, service downtime, number of re-deployments, time spent for each re-deployment, SLA improvement after each redeployment and time between re-deployments	ADAPT Violations Handler already accepts alerts (violations) in this M24 release; support for the required metrics on re-deployments will be analyzed for the next release
ARR09 Automatic deployment tools	ADAPT should include tools based on the container technology in order to automate all the deployments.	ADAPT is already based on containers and provides deployment automation, given a proper Application Description

<sup>11</sup> For example HPE OneSphere ([www.hpeonesphere.com](http://www.hpeonesphere.com)) for multi-cloud management

Req. ID and Title	Description	Impact on ADAPT
EXPR01 Requirements tracking	Take into account the client's requirements during the project's lifecycle so developers can respond to them efficiently.	Client's requirements are expressed in DECIDE through the NFR Editor and MCSLA Editor; deployment schemas to satisfy them are produced by OPTIMUS and ACSmI; ADAPT deploys the schema, collects any violations to the indicated MCSLA and triggers a redeployment to cope with them
EXPR02 Deployment configurations	Reduce deployment time and costs by easing the deployment of applications in multi-cloud environments.	Multi-cloud deployment was already available in M12 release of ADAPT
EXPR03 Environment replication	Replicate the production environment easily to allow the execution of tests on real data without huge hardware costs.	A further deployment replicating the production environment can be done submitting to ADAPT the same Application Description; to obtain that the new environment executes in parallel on the same production data, a further application component would be needed to duplicate input data from production to the test environment
EXPR05 Maintenance	Do maintenance tasks without shutting down the whole system.	The requirement is quite generic, but one possibility to satisfy it is that the application supports replicated stateless services; support for deploying replicated containers will be analyzed for the next release
EXPR06 Service monitoring	React quickly to errors in the application by monitoring microservices and managing alerts.	The M24 release ADAPT is planned to monitor microservices and react to alerts (violations); how quick the reaction will be is still to be measured
EXPR07 Unsupervised operation	Have the applications to be reconfigured automatically in case of violations of the NFRs so that the client receives a better service.	The M24 release of ADAPT is planned to automatically react to violations: in this case a new redeployment will be triggered by asking OPTIMUS for a new deployment schema

### 3 High Level ADAPT Functionalities

This section reports the high level functionalities identified so far for the DECIDE ADAPT tool.

#### 3.1 Updated ADAPT UML Use Cases

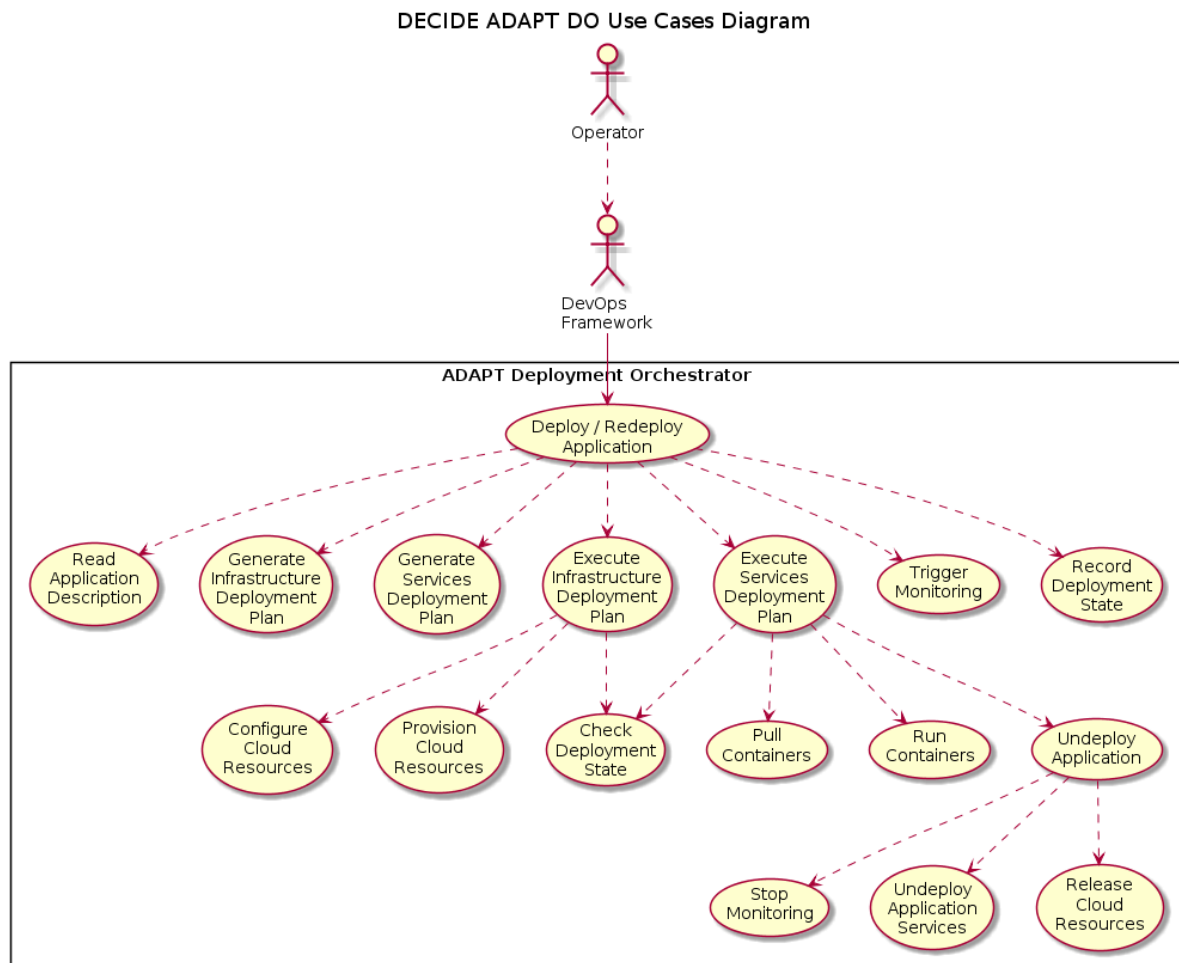
The following sub-sections describe the current view of both implemented and planned functionalities for the three ADAPT components: Deployment Orchestrator (DO), Monitoring Manager (MM), and Violations Handler (VH).

##### 3.1.1 ADAPT DO Use Cases

Requirements planned to be satisfied by ADAPT DO in Y2 are the following:

- WP4-MR1 (Semi-automatic adaptation and redeployment)
- WP4-MR8 (Low technological risk = automatic redeployment)
- WP4-MR11 (Application Description drives ADAPT behavior)
- WP4-MR15 (Business continuity)
- WP4-MR16 (Continuous deployment)

The ADAPT DO functionalities supporting the above requirements are described by the UML use cases detailed in **Figure 1**. The Year 1 use cases, already documented in D4.1 [3], are superseded by the new ones.



**Figure 1.** ADAPT Deployment Orchestrator Use Cases diagram



The first Agent in **Figure 1**, the Operator, actually indicates a role: the person can be a Developer or better a DevOps team member taking care of deploying and operating the application.

### **UCDO201 – Deploy / Redeploy Application**

This is the main use case of ADAPT DO, representing both its deploy and redeploy functionalities. A single use case is used since both functionalities can be invoked using the same (REST API) call. The system keeps a deployment state: if the current deployment state is not empty then the functionality is a redeployment, otherwise it is an initial deployment. The steps executed in both cases are the following:

- Read Application Description (UCDO202)
- Generate Infrastructure Deployment Plan (UCDO203)
- Generate Services Deployment Plan (UCDO204)
- Execute Infrastructure Deployment Plan (UCDO205)
- Execute Services Deployment Plan (UCDO209)
- Trigger Monitoring (UCDO216)
- Record Deployment State (UCDO217)

### **UCDO202 – Read Application Description**

The Application Description (AD) is read from the given git repository and parsed to extract all the information needed to guide the (re)deployment process.

### **UCDO203 – Generate Infrastructure Deployment Plan**

From the parsed AD, ADAPT DO generates an infrastructure deployment plan. In case of a redeployment, the plan specifies to transform the current deployment state into the target state indicated in the AD. The plan should specify that the current infrastructure is dismantled only after the execution of the services deployment plan (UCDO209), to avoid impact on business continuity.

### **UCDO204 – Generate Services Deployment Plan**

From the parsed AD, ADAPT DO generates a services deployment plan. In case of a redeployment, the plan specifies to transform the current services deployment state into the target state indicated in the AD (if it's an initial deployment the current deployment state is empty). The plan should specify that the current services are stopped only after the startup of all the new services. The application under deployment should provide mechanisms exploiting this ordering, such as those suggested in section 3.2 (and section 3.4 of D4.1 [3]), to avoid impacting business continuity.

### **UCDO205 – Execute Infrastructure Deployment Plan**

This use case executes the infrastructure deployment plan generated by use case UCDO203. Typically the first action executed is a check of the current deployment state (UCDO206); then the system provisions any new cloud resource needed to reach the target state (UCDO207), and configures the provisioned cloud resources (UCDO208).

### **UCDO206 – Check Deployment State**

This use case checks the current deployment state for the given application. From the difference between the target state and the current one, the system infers the actions needed to reach the target state.

### **UCDO207 – Provision Cloud Resources**

This use case provisions the needed cloud resources on their respective provider, as indicated in the Application Description. This provisioning is obtained by invoking the ACSmI interface to obtain / release resources. The same interface allows both to create cloud resources (Virtual Machines) and to define firewall rules to protect the created resources.

#### **UCDO208 – Configure Cloud Resources**

This use case configures the created cloud resources to allow hosting the services constituting the given application. Typically this configuration means installing and configuring the platform (e.g. Docker<sup>2</sup>) that will allow the deployment of the application's containers. The configuration may also include the installation of any further components needed in the operational phase (e.g. Consul<sup>3</sup>), and any network configuration (e.g. VPN or IPsec overlay) aimed at increasing the security level of the application to be deployed.

#### **UCDO209 – Execute Services Deployment Plan**

This use case executes the services deployment plan generated by use case UCDO204. Also in this case the first step is a check of the current deployment state (UCDO206); then the system pulls from their respective registry all the containers to be deployed (UCDO210) and runs them (UCDO211). Finally, in case of a redeployment, when the new application is running the previous application configuration is undeployed (UCDO212).

#### **UCDO210 – Pull Containers**

Cloud-native applications, the target of ADAPT deployment work, are composed by one or more containers, each possibly located in a different container registry, as described by the Application Description. This use case collects all the container images needed to run the application, downloading them in the local cache of the nodes / providers where they will be deployed. The download (*pull*) operation compares the version available in cache, if any, with the version in the registry: if they are different the new image is downloaded. Typically for a new deployment all the images are downloaded, but in the case of a redeployment (on the same node) only what has changed must be transferred.

#### **UCDO211 – Run Containers**

This use case starts the containers that are part of the application on their respective nodes / providers, as indicated by the Application Description. A container can be run on a node only when its image is available on that node, therefore the execution of use case UCDO210 (Pull containers) is a prerequisite for this one.

#### **UCDO212 – Undeploy Application**

This use case is invoked in case of a redeployment, to undeploy the previous configuration of a given application. The first step is to stop monitoring the old application's services (UCDO213), then those services are undeployed (UCDO214) and any unused cloud resources are released (UCDO215).

#### **UCDO213 – Stop Monitoring**

In case of a redeployment, this use case stops the monitoring activities on the old application's services by calling the related interface of ADAPT MM (use case UCMM05).

---

<sup>2</sup> <https://www.docker.com/>

<sup>3</sup> <https://www.consul.io/>

### UCDO214 – Undeploy Application Services

In case of a redeployment, this use case undeploys the old application's services by stopping and then removing the related containers. To ensure a graceful service shutdown when the container is stopped is a developer responsibility. To do that, the service should properly handle the SIGTERM signal and the container should be built in such a way that any signal sent to it reaches the service. See [6] for more information on how to gracefully stop Docker containers.

### UCDO215 – Release Cloud Resources

When cloud resources are not used it is good practice to release them, in order to both save money and keep the configuration clean. This use case releases unused cloud resources, typically stopping nodes (Virtual Machines) where no more services are running.

### UCDO216 – Trigger Monitoring

This use case triggers monitoring activities on the newly deployed application's services by calling the related interface of ADAPT MM (use case UCMM01).

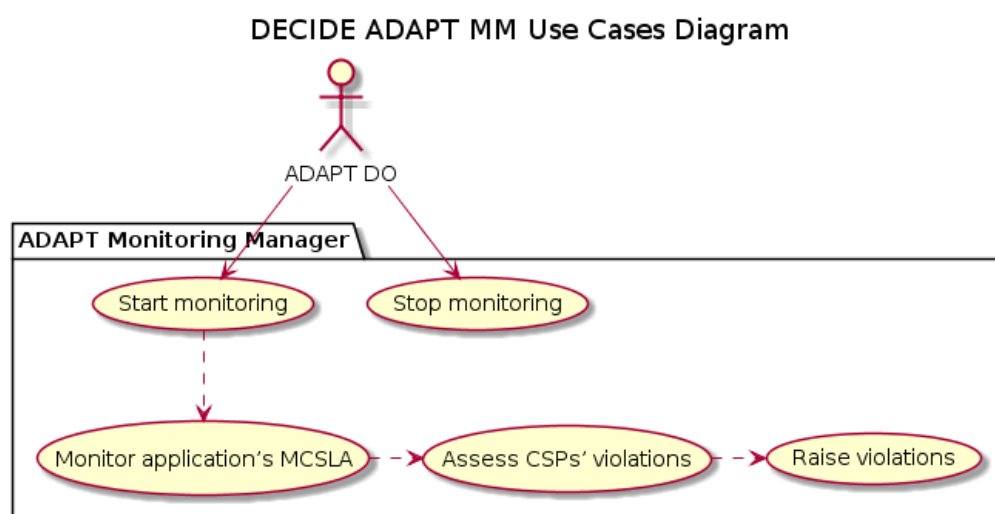
### UCDO217 – Record Deployment State

This use case records the current deployment state for the given application. The deployment state includes both information on infrastructure resources in use for all the providers involved in the current application deployment, and information on services deployed for the same application on those providers. To increase ADAPT reliability, the deployment state should be recorded on an external storage service which could survive any ADAPT restart (see [7] about Terraform remote state).

## 3.1.2 ADAPT MM Use Cases

ADAPT Monitoring Manager (MM) monitors the multi-cloud application deployed by ADAPT DO at real time with respect to the NFRs defined in the MCSLA, assesses them against the SLOs defined in the MCSLA and if a violation occurs, it informs the VH so that the corresponding actions are launched. ADAPT MM also requests ACSmI to start monitoring the resources (Cloud Services) where the different components of the multicloud application are deployed.

The **Figure 2** below shows the ADAPT MM's use cases:



**Figure 2.** DECIDE ADAPT MM Use Cases Diagram

### UCMM01 – Start monitoring

Once the deployment of the application is triggered by the ADAPT Deployment Orchestrator, the monitoring request occurs. The ADAPT Monitoring Manager will receive the request to start the monitoring of the application. The 'Start monitoring' process will include the following actions:

- Request the SLOs, the NFRs and the MCSLA values to be monitored. These values are obtained from the application description through the application controller.
- Establish the (new) measurements to be gathered both at software level and at resource level. If any of the established metrics is already being monitored, only the new ones will be configured and deployed for monitoring.
- Deploy/configure the data collection means (i.e. agents, DB, etc.) along with the multi-cloud application components.
- Launch the Assess CSP's violation process.
- Configure the monitoring dashboards with the selected measures and thresholds.

### UCMM02 – Monitor application's MCSLA

After the monitoring of the application has been configured, the monitoring process starts. This implies the update of the application monitoring status in the Application Description through the Application Controller. The deployed agents continuously provide measurements to be stored (sampling of the metrics). The ADAPT MM components aggregate the stored data to create the required values to be compared with the established SLOs and MCSLA thresholds. The monitored measures are provided to the multi-cloud application owner through the UI.

When an application is not being monitored any more, the ADAPT MM component will mark the corresponding registries in the Application Description as not being monitored and the monitoring resources will be stopped (that is, monitoring agents will not send monitoring information to the data base). The corresponding measures are not being shown any more in the user interface.

### UCMM03 – Assess CSPs' violations

This functionality covers the request for start/stop the assessing of the collected metrics of the CSPs.

### UCMM04 – Raise violations

The measured and compiled data is continuously being assessed against the SLOs and thresholds. When a violation is detected an alert is sent by ADAPT MM to the operator. At the same time the violation detection is sent to the Violations Handler so that the corresponding actions are triggered.

### UCMM05 – Stop monitoring

When ADAPT MM receives the request to stop the monitoring for any of the components of the application, the following actions need to be performed:

- Stop the monitoring agents and release the corresponding resources.
- Stop the calculation and assessment of the aggregated measurements.
- Send a request to ACSmI for stopping assessing the CSP metrics.

## 3.1.3 ADAPT VH Use Cases

ADAPT Violations Handler (VH) is a component that is in charge of receiving and processing violations, which are detected by ACSmI's and ADAPT's monitoring components. When a violation occurs, the Violations Handler notifies the operator and, depending on the technological risk of the application,

automatically redeploys the application or requests the operator's permission to perform a redeployment.

The Figure 3 below shows the VH's use cases:

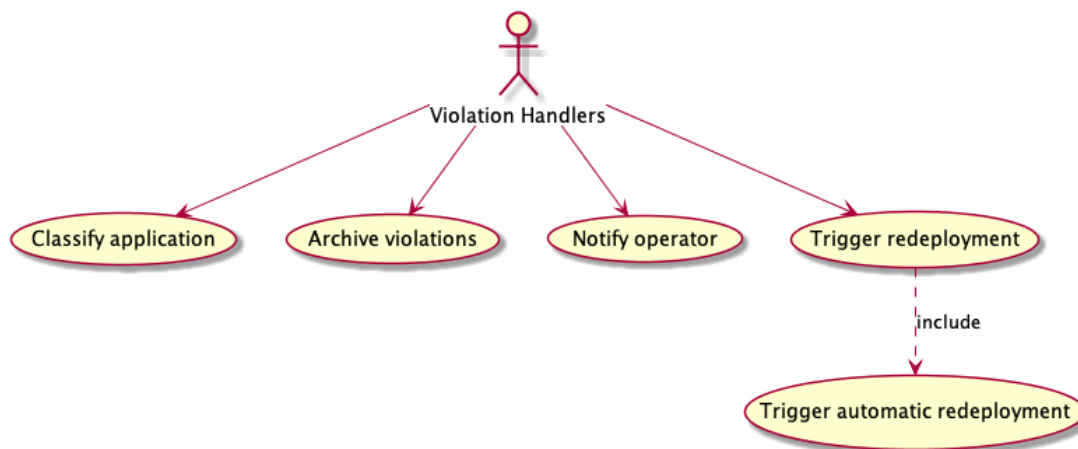


Figure 3. Violations Handler's use cases

#### UCVH01 - Classify application

The Violations Handler must obtain the technological risk of the application from the Application Description. Depending on its value, it will automatically redeploy the application (if the technological risk is low) or ask the operator to choose a new deployment configuration before redeploying the application (if the technological risk is high).

#### UCVH02 - Archive violations

The VH is in charge of storing the received violations, which will be visualized within ADAPT Monitoring Manager. The violations will be stored either in the Application Description or in a local database.

#### UCVH03 - Notify operator

When the VH receives a violation, the operator will be notified by means of an email message.

#### UCVH04 - Trigger redeployment

If the application is classified as “low technological risk”, upon receiving a violation, the VH will automatically trigger a new simulation through OPTIMUS, which will, in turn, instruct ADAPT to deploy the application according to the newly found deployment configuration.

#### UCVH05 - Trigger automatic redeployment

If the application is classified as “high technological risk”, upon receiving a violation, the VH will give the operator the possibility to choose a new deployment configuration before redeploying the application.

## 3.2 Monitoring scalability

Real-time monitoring produces a lot of data that needs to be kept in order to analyse it or perform further operations. Usually, monitoring systems are based on a centralized daemon which gathers the information from the distributed agents running on every node.

In the case of ADAPT MM, a metric collection daemon is used (Telegraf<sup>4</sup>) to collect metrics from a multi-cloud Application and send them to a database. It is plugin-based both to retrieve and provision the data to the database.

ADAPT MM needs to have available all the metrics and data collected by the monitoring agent. For this a time series data base has been selected. A Time Series Database (TSDB) is a database optimized for time-stamped or time series data, that is arrays of numbers indexed by time (a datetime or a datetime range). Time series may represent measurements or events which can then be tracked, monitored, and aggregated over time [8]. Software with complex logic or business rules and high transaction volume for time series data may not be practical with traditional relational database management systems. Flat file databases are not a viable option either, if the data and transaction volume reaches a maximum threshold determined by the capacity of individual servers (processing power and storage capacity) [9].

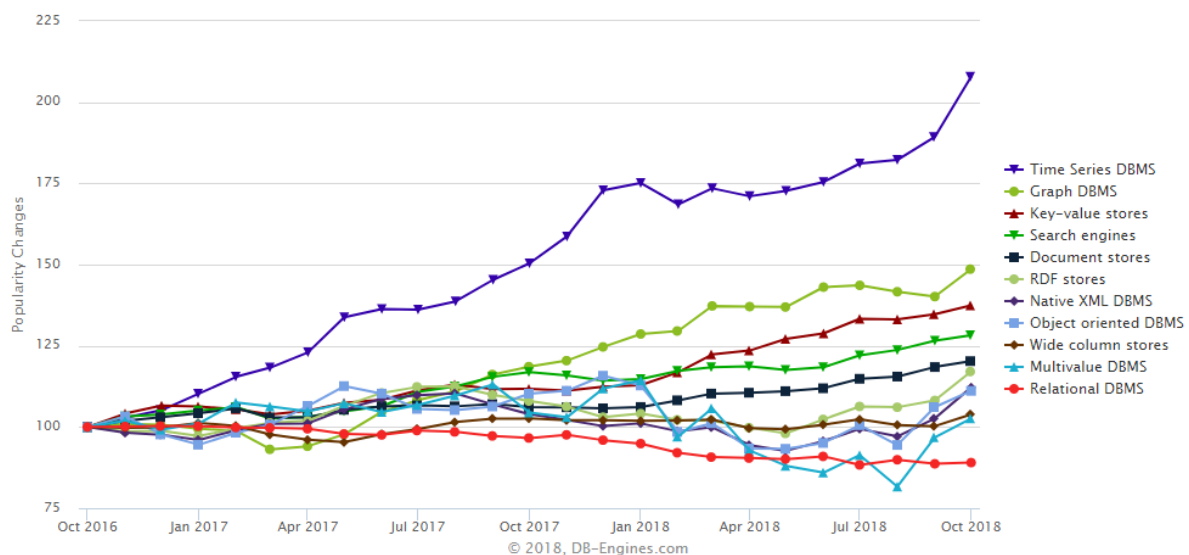
To provide value, this data requires aggregation and analysis.

A Time Series Database (TSDB) is specifically built for handling a high amount of metrics or measurements that are time-stamped. A TSDB is optimized to track change over time. Time series data is different from other data workloads with respect to data lifecycle management, summarization, and large range scans of many records.

Time Series Databases are very different from other databases for some key architectural design properties, such as: data lifecycle management, time-stamp data compression and storage, data summarization, time series aware queries, and ability to handle large scans of many records [8].

Data Bases expert , such as the DB-engines initiative [10] have published several analysis on the growth of Time Series Data Bases popularity in the last years (see **Figure 4**):

#### Trend of the last 24 months



**Figure 4.** Database management systems ranking according to popularity (source DB-engine [11])

As seen in the previous figure Time Series Database are the fastest growing segment in the database industry.

<sup>4</sup> <https://www.influxdata.com/time-series-platform/telegraf/>

DB-Engines also ranks different Time Series databases based on search engine popularity, social media mentions, job postings, and technical discussion volume [12]. This ranking is updated every month.

26 systems in ranking, October 2018















Rank			DBMS	Database Model	Score		
Oct 2018	Sep 2018	Oct 2017			Oct 2018	Sep 2018	Oct 2017
1.	1.	1.	InfluxDB 	Time Series DBMS	12.98	+1.19	+4.28
2.	2.	 5.	Kdb+ 	Multi-model 	4.37	+0.50	+2.54
3.	3.	3.	Graphite	Time Series DBMS	2.81	+0.11	+0.05
4.	4.	 2.	RRDtool	Time Series DBMS	2.67	+0.12	-0.44
5.	5.	 4.	OpenTSDB	Time Series DBMS	1.96	+0.16	+0.10
6.	6.	 7.	Prometheus	Time Series DBMS	1.71	+0.12	+0.98
7.	7.	 6.	Druid	Time Series DBMS	1.32	+0.11	+0.32
8.	 9.		TimescaleDB	Time Series DBMS	0.57	+0.09	
9.	 8.	 8.	KairosDB	Time Series DBMS	0.53	+0.00	+0.05
10.	10.	 9.	eXtremeDB 	Multi-model 	0.30	-0.02	-0.02

Figure 5. Top 10 Time Series databases and their historical changes (source DB-engine [12])

Based on this information and other analysis [8], we have selected InfluxDB as the Time Series Data Base for ADAPT MM.

Selecting a Time Series Data Base provides by itself the certain scalability characteristic due to the key architectural design properties that make them very different from other databases.

A typical query with a Time Series Database is to request a summary of data over a large time period. Such query requires to go over a range of data points and perform some computation of a metric over the period, say the last six months. This type of query is very difficult to optimize with a distributed key value store. TSDB's are optimized for this kind of data gathering. Another example can be the need to keep high precision data for a short time interval, to be aggregated and then downsampled into longer term trend data. This means that every data point inserted into the database will have to be deleted when its time period expires. This kind of data lifecycle management is difficult to implement using regular databases, whereas Time Series Databases are optimized for it [8].

One of the advantages of Time Series Databases is their massive scalability and performance [13] as it enables applications to scale up to time series containing millions of data points in a continuous flow and still perform real-time analysis. The only downside of InfluxDB is that scalability (i.e. clustering) is a premium feature not available in the open source version, but this may not be a big issue since the price is not prohibiting even for smaller companies [14].



## 4 DECIDE ADAPT Updated Architecture

A logical view of DECIDE ADAPT architecture is shown in the following Figure 6 along with the external integration interfaces.

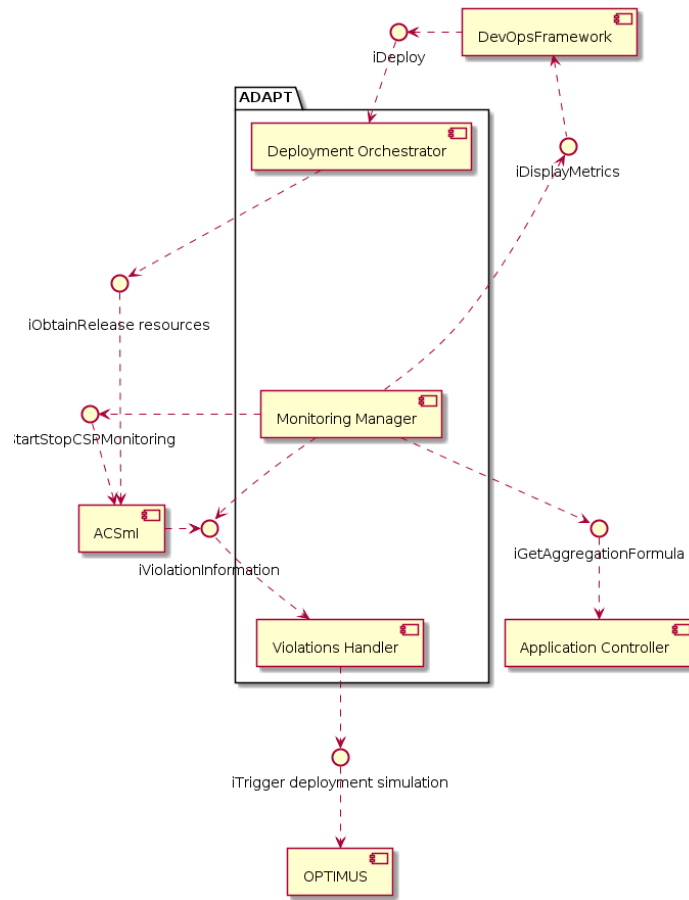


Figure 6. DECIDE ADAPT logical architecture and external interfaces

The main components of DECIDE ADAPT are the following. The components are the same of the first release, but with increased functionalities and better integration.

**Deployment Orchestrator.** The Deployment Orchestrator (DO) is in charge of orchestrating the deployment / redeployment lifecycle for user applications and their components. Details on the current version of Deployment Orchestrator can be found in section 4.1 and in deliverable D4.5 [1].

**Monitoring Manager.** The Monitoring Manager (MM) controls the monitoring functionality for the application, according to its defined MCSLA (Multi-Cloud SLA), and raises any related violations, including those for the CSPs where the application is deployed on. Details on the current version of Monitoring Manager can be found in section 4.2 and in deliverable D4.8 [2].

**Violations Handler.** The Violations Handler (VH) will handle both violation raised by the Monitoring Manager and by ACSml, regarding respectively the application MCSLA or the CSPs' NFRs. Violation handling may lead both to alerting the operator and to contacting OPTIMUS to trigger a new re-deployment simulation for the application, thus starting a readaptation process. Details on the current version of Violations Handler can be found in section 4.3 and in deliverable D4.8 [2].

Figure 6 also shows the current view of the integration interfaces between ADAPT and other DECIDE tools, listed below.



**iDeploy.** This interface allows the DevOps Framework to ask ADAPT DO for the deployment of an application. The related Application Description is obtained from the Git repository indicated in the deployment request. The same interface is called both for a new deployment and in case of a redeployment.

**iObtainReleaseResources.** This ACSml interface is called by the Deployment Orchestrator when it needs to provision resources on a specific cloud provider as indicated by the Application Description. The same interface is also called during a redeployment to release allocated resources which are no longer needed. The first version of this interface was already integrated in the first year; in its second version the interface also allows provisioning firewall rules, and it is currently being extended to provision resources from a private cloud.

**iViolationInformation.** This interface, exported by Violations Handler, was an internal interface called only by Monitoring Manager in the first DECIDE release to report MCSLA violations. In this updated second release the same interface is also called by ACSml to report CSP violations.

**iStartStopCSPMonitoring.** This new ACSml interface is called by the Monitoring Manager to start monitoring the CSPs involved in the current deployment. The same interface is called to stop monitoring those CSPs that are not part of a redeployment.

**iDisplayMetrics.** This DevOps Framework interface is called by the Monitoring Manager to display the collected monitoring metrics.

**iGetAggregationFormula.** This Application Controller interface is called by the Monitoring Manager to obtain the right formula to aggregate metrics for the current application as required in the MCSLA. The aggregation formula depends on the specific metric and may depend also on the selected deployment schema/pattern indicated the Application Description.

**iTriggerDeploymentSimulation.** This OPTIMUS interface is called by the Violations Handler to trigger a new deployment simulation, when a redeployment is needed to cope with a violation.

## 4.1 ADAPT DO architecture

DECIDE ADAPT Deployment Orchestrator (DO) gets as input the Application Description and is in charge of deploying and undeploying the application and its components on the indicated cloud providers. Infrastructure resources from the cloud providers are obtained and released through ACSml.

Application adaptation to cope with SLA violations is obtained by redeploying the application according to the new configuration calculated by OPTIMUS.

A redeployment is obtained invoking the same iDeploy REST interface for the same application with a different Application Description. ADAPT DO keeps the deployment state for the application and if it is not empty then it knows the functionality is a redeployment, otherwise it is an initial deployment.

The internal architecture of ADAPT DO is shown in the following Figure 7.

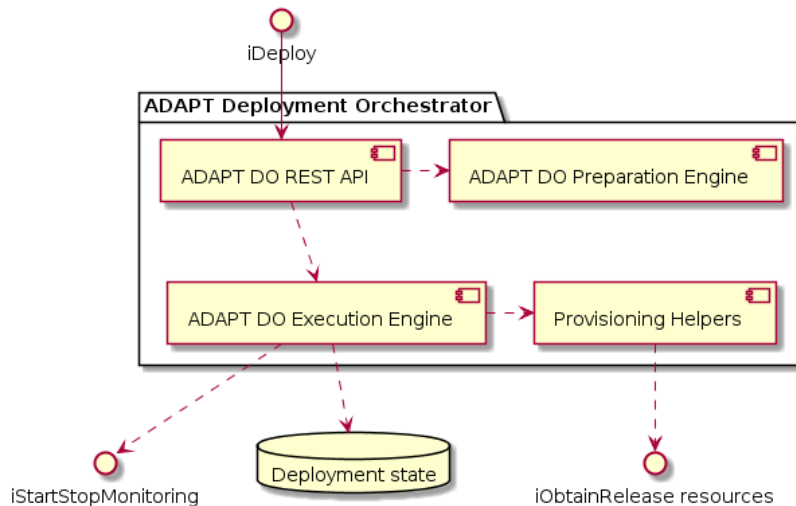


Figure 7. ADAPT Deployment Orchestrator components

**ADAPT DO REST API.** This is the main component of the ADAPT Deeployment Orchestrator. It orchestrates the needed deployment steps, manages the generation of the deployment plan, manages the plugins of the ADAPT DO Execution Engine and activates it.

**ADAPT DO Preparation Engine.** This component, based on the information included in the Application Description, generates all the needed deployment plans and scripts to be executed by the ADAPT DO Execution Engine.

**ADAPT DO Execution Engine.** This component creates an initial deployment state for each environment, checks the generated deployment plan and finally executes it. A deployment plan defines the infrastructure resources needed for deployment along with their expected configuration and any command script to bootstrap them.

**Provisioning Helpers.** The need for Helpers in ADAPT has been identified so far only in the Deployment phase and this component has therefore been included within DO itself. The Provisioning Helpers support DO in provisioning the resources required for deployment by interfacing with an external provisioning component, which is ACSmI for all the supported clouds, but may also be something different in specific cases. Currently ADAPT DO is being extended for direct access to OpenStack-based private clouds to enable the hybrid deployment scenarios described in section 6.2. More information on the Helpers can be found in deliverable D4.5 [1].

## 4.2 ADAPT MM architecture

This section explains the architecture of ADAPT Monitoring Manager (MM). It presents an updated version of the D4.1 section 4.1 [3]. This version corresponds to the M24 prototype of ADAPT MM.

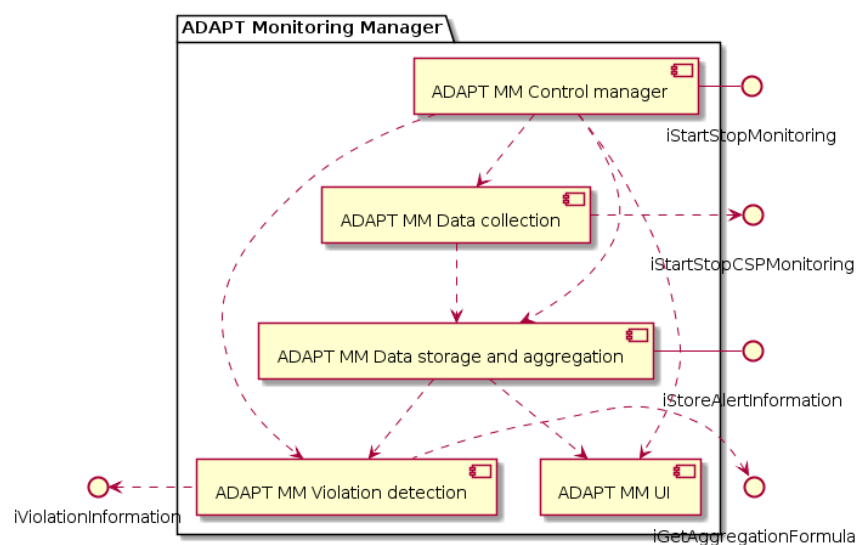
The DECIDE ADAPT MM components monitor the deployed multi-cloud application and verify that the non-functional requirements and the SLOs (defined as part of the multi-cloud SLA) are being fulfilled. ADAPT MM also informs ACSmI to start or stop the assessment of the SLOs corresponding to CSPs related metrics.

If a violation of any of the NFRs or SLOs is detected, ADAPT MM components will inform the Violation Handler component which will generate the proper actions depending on each situation and context. If the application is low technology risk, the “adaptation” process will be launched, through the Violation Handler component.

The main functionalities of the ADAPT MM components are:

- Collect data from the deployed multi-cloud application and the underlying cloud resources: The ADAPT MM component needs to get the data from the deployed components and their NFRs as well as from the underlying cloud services to collect data from their service level metrics at real time. The data related to the cloud services will be then assessed by ACSml.
- Store the data collected from the deployed multi-cloud application: to assess the required working conditions of the multi-cloud application ADAPT MM will process and combine metrics measured in real time. These time series metrics will be stored for further analysis of the data.
- Create the aggregated data to be assessed: from the raw metrics, the ADAPT MM component will need to create aggregated data for assessing the violations of the SLOs.
- Visualize the measurements: DECIDE ADAPT MM will provide the user with an interface to visualize the monitored data, from the multi-cloud application and from the underlying cloud services. DECIDE ADAPT MM will also provide monitoring information of the violation occurred.
- Detect when a violation occurs: DECIDE ADAPT will detect violations on the working conditions (NFRs/SLOs) at operation time and launch the “adaptation” process. The adaptation process launched will include the actions related to stop the monitoring of the previous components and resources. Monitoring of the resources will not be stopped until the resources are undeployed.

These functionalities will be covered by the following sub-components inside ADAPT MM, also shown in Figure 8.



**Figure 8.** ADAPT Monitoring Manager (MM) internal component diagram

- **ADAPT MM Control manager:** This sub-component is in charge of managing the different processes and requests that need to be triggered in each of the other sub-components of ADAPT MM. ADAPT MM Control manager will launch the following processes:
  - Start monitoring
  - Stop monitoring
  - Monitor application’s MCSLA
  - Raise violations
- **ADAPT MM Data collection:** This sub-component will collect the data from different sources. On one hand, it will collect data from the resources where the different microservices are

deployed. On the other hand, it will collect data (metrics) from the microservices themselves. The Data collection sub-component will be based on agents deployed within the different components and cloud resources. The agents will be pre-defined and pre-implemented to be installed when deploying the multi-cloud application. The ADAPT MM Data collection sub-component will receive the requests from the ADAPT Deployment Orchestrator, both for starting and stopping the monitoring of a multi-cloud application.

- ADAPT MM Data storage and aggregation: This sub-component will be in charge of storing the data collected from the ADAPT MM Data collection and aggregating it to create the actual measures that will be assessed by ADAPT MM Violation detection.
- ADAPT MM Violation detection: This component will be in charge of assessing that the required working conditions are or are not being met. The ADAPT MM Violation detection will need to get the SLOs for the different metrics and the related aggregation formula from the Application Controller.
- ADAPT MM UI: This is the graphical user interface for the ADAPT MM component. It will provide the means for the operator of the multi-cloud application to visualize the metrics at run-time and the information from the violations occurred.



Figure 9. Excerpts of ADAPT MM UI

The component diagram in Figure 10 shows ADAPT Monitoring Manager's (MM) external interfaces.

- ADAPT MM will gather information from the Application Description through the Application Controller (the threshold values for the different metrics to be assessed plus the aggregation formula), and from the Violations Handler (historical information about the alerts).
- ADAPT MM will provide information to the Violations Handler about any violation occurred.
- ADAPT MM will request ACSml to start the monitoring of the cloud services where the components of the multi-cloud applications are deployed.
- ADAPT MM will receive requests from ADAPT Deployment Orchestrator to start/stop the monitoring.

In the following Figure 10 these communications are shown:

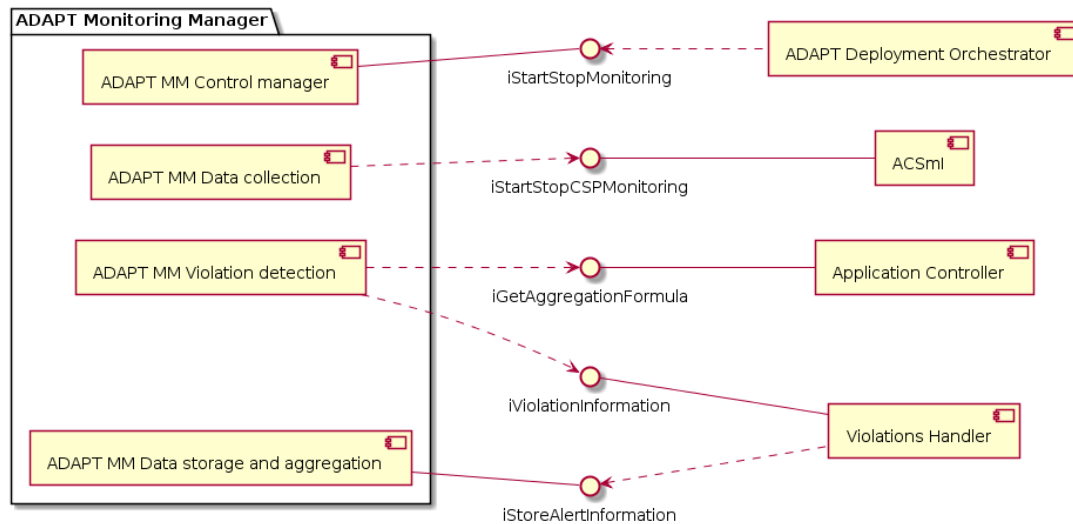


Figure 10. ADAPT Monitoring Manager external component diagram

### 4.3 ADAPT VH architecture

As it has been stated before, the Violations Handler is in charge of processing the violations received from ACSml's and ADAPT's monitoring components to carry out an appropriate action, which depends on the application's technological risk.

The architecture of this component is shown in the Figure 11 below:

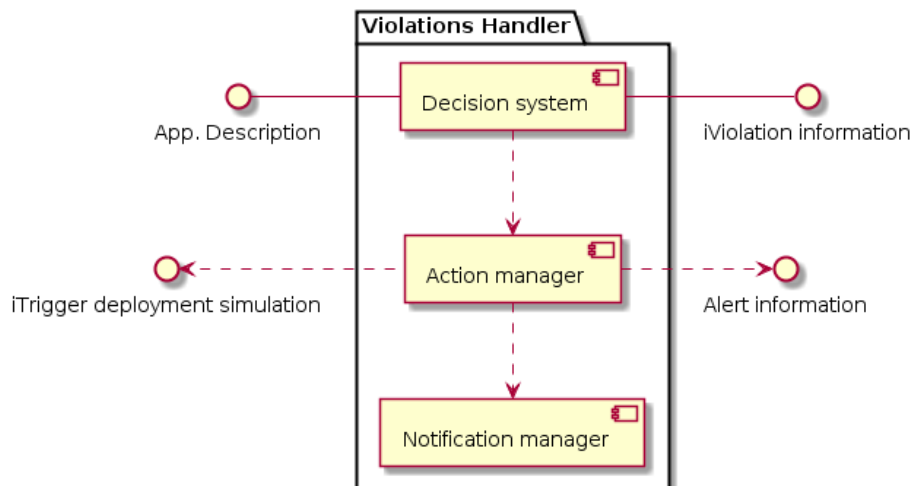


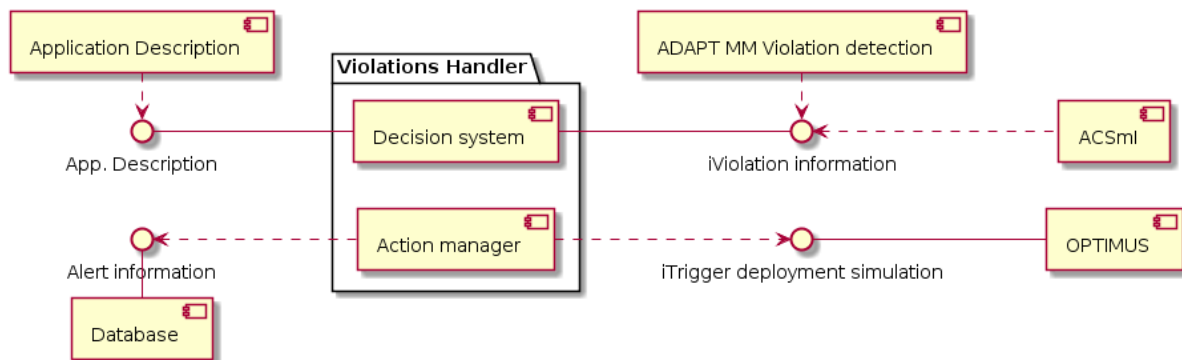
Figure 11. ADAPT Violations Handler component diagram

The *Decision system* receives the violation and requests the Application Description the application's technological risk. Depending on this variable, it will instruct the *Action manager* to perform the corresponding actions:

- Instruct the *Notifications manager* to notify the operator. The notification will be done through an email that will contain information regarding the alert
- Send a message to OPTIMUS, containing the necessary data to start a new simulation process

The *Action manager* will also store the received alerts either in the application description or in a local database to keep a history of alerts and to visualize them from ADAPT's UI.

Figure 12 shows these external interfaces:



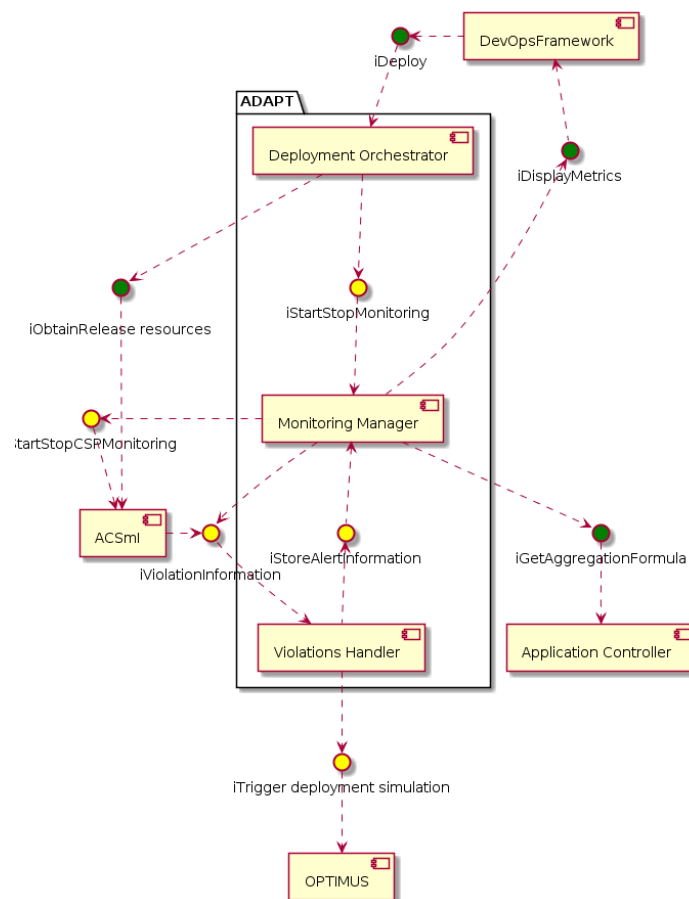
**Figure 12.** ADAPT Violations Handler external interfaces diagram

## 5 DECIDE ADAPT Interfaces

DECIDE ADAPT integration is on-going and interfaces will be documented in more detail in the implementation deliverables, namely, D4.5 [1] and D4.8 [2]. This section shows an overview of the current status.

### 5.1 ADAPT integration

This section reports about all the integration interfaces for ADAPT, both internal and external, and shows their implementation and integration status.



**Figure 13.** Internal and external ADAPT interfaces and their status

The external integration interfaces have already been described in section 4. The internal interfaces are detailed next.

**iStartStopMonitoring.** This Monitoring Manager interface is called by the Deployment Orchestrator to start monitoring after an application’s deployment; the same interface is called also to stop monitoring the old application’s components in case of a redeployment.

**iStoreAlertInformation.** In the original plan VH would call this interface to record alert information. In the last version this interface is going to be removed, since VH keeps alert history in a local database

Figure 13 also shows, color coded, the implementation and integration status of each interface at the time of writing (M24). The current status of each interface is also indicated in the following Table 4.

**Table 4.** Current status of ADAPT interfaces

Interface	Designed	Coded / Tested	Integrated	Comments
<b>iDeploy</b>	Yes	Yes	Yes	
<b>iDisplayMetrics</b>	Yes	Yes	Yes	URL written in the App Description, DevOps Framework read and loads it
<b>iGetAggregationFormula</b>	Yes	Yes	Yes	The interface calls a library that reads ISO standard metrics description language from Application Description and returns the aggregation formula
<b>iObtainReleaseResources</b>	Yes	Yes	Yes	V1 integrated, V2 (extension for firewall rules) implemented, V3 (private cloud) in development
<b>iStartStopCSPMonitoring</b>	Yes	Yes	No	Integration approach recently revised and interface redesigned
<b>iStartStopMonitoring</b>	Yes	Yes	No	Integration ongoing
<b>iStoreAlertInformation</b>	Yes	Yes	No	In the last version this interface is going to be removed, since VH keeps the alert history in a local database
<b>iTriggerDeploymentSimulation</b>	Yes	Yes	No	Two cases have been identified, a simpler one (v1) and a complex one (v2); v1 implemented and under integration test, v2 in design phase
<b>iViolationInformation</b>	Yes	Yes	No	Design recently revised, implementation started

## 5.2 Application Description

The Application Description (AD) is the main way of exchanging complex information between all DECIDE tools. This section summarizes the current status of information exchanged by ADAPT through the AD. The last version of the whole list of AD fields is documented in appendix A of deliverable D2.5 [5].



Writer	Fields	Description	Reader
DevOps Framework	consulJoinIp	Address of the master Consul (microservice registry) node; it may be the address of the node running ADAPT	ADAPT DO
DevOps Framework	highTechnologicalRisk	If true the application has high technological risk: confirmation for (re)deployment is needed	ADAPT VH
ADAPT MM	monitoring.status, monitoring.urls	Triggers the display of monitoring metrics in the DevOps Framework	DevOps Framework
ADAPT DO	virtualMachines. dockerHostPublicIp	IP address of the VM / Docker node	ADAPT MM
ACSml Contracting	virtualMachines.*	Info for provisioning the VMs which will host the containers	ADAPT DO
ACSml Contracting	containers. dockerHostName	Indicates the VM on which ADAPT should deploy the container	ADAPT DO
Developer / DevOps Framework	containers.*	Description and information to deploy each container	ADAPT DO
ADAPT DO	applicationInstanceId	Unique Id for this deployed application	ADAPT MM
MCSLA Editor	mcsla.sla.objectives.*	SLA with application metrics to be monitored by ADAPM MM	ADAPT MM
MCSLA Editor	mcsla.csSla.objectives.*	SLA with infrastructure metrics to be monitored by ACSml; violations can be sent to ADAPT VH about these metrics	ACSml

## 6 ADAPT deployment scenarios

An initial ADAPT deployment scenario has already been implemented for both integration and demo purposes, as a centralized deployment in the integration environment. A different deployment option, involving one ADAPT instance for each application, has been explored and then abandoned for two main reasons: to avoid moving historical monitoring data around for each redeployment and to avoid developers to provision (and pay for) further resources for each application just to run ADAPT. The current idea is to have a central deployment for ADAPT which will handle multiple users and multiple applications for each user. Further requirements to accommodate such a multi-tenant deployment, such as scalability and security, will be explored in the next year.

This section discusses multiple cases of central deployment scenarios: the testing and integration deployment platform used for DECIDE development, a possible ADAPT-as-a-Service scenario, and a hybrid scenario to support private clouds as well as public ones.

### 6.1 ADAPT integration platform

The workflow within DECIDE framework requires multiple communication between all the tools that participate in the process. Thus, an integration environment has been created under the same virtual machine hosted by a public cloud provider (AIMES infrastructure), where all the tools are deployed following CI and CD pipelines during their development phase. In this scenario, ADAPT is executed as an standalone container deployed in the private cloud. Therefore every tool that needs to consume the ADAPT microservice is able to discover it within the same infrastructure.

This scenario proposes an integration approach with all the KRs accessible each other. The ADAPT interface is directly consumed from the DevOps Framework, which offers a web client from where the user can consume the ADAPT service easily. This environment is mainly oriented for testing the integration between the components, and evaluate the impact of ADAPT in the proposed use cases, considering it as a staging scenario for continuous integration and development, and never as a production platform.

The integration scenario which is currently being implemented involves running ADAPT in the AIMES cloud infrastructure and provisioning public cloud resources through ACSml, as shown in Figure 14.

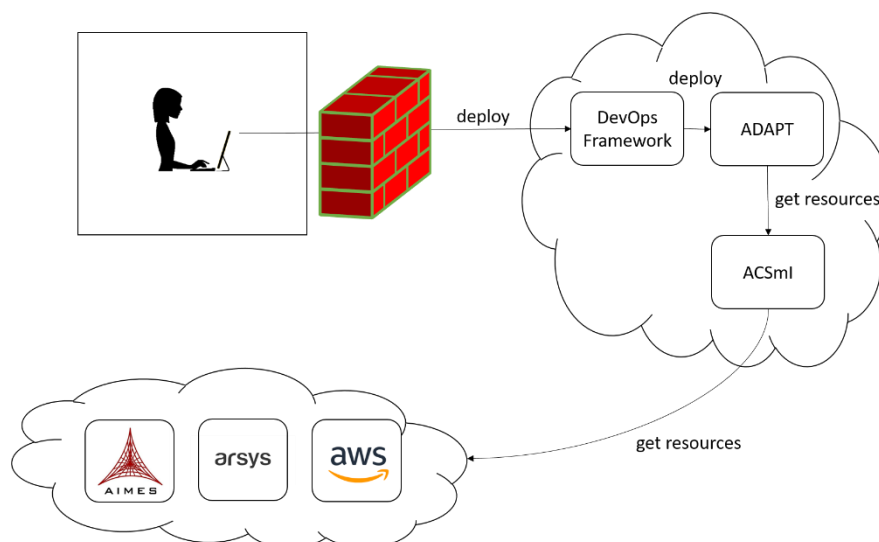


Figure 14. ADAPT integration platform scenario

## 6.2 ADAPT Hybrid scenario

ADAPT is currently being extended to support also an Hybrid Cloud scenario. As by multi-cloud we mean the use of multiple clouds at the same time for the same application, by Hybrid Cloud we mean the use of both private and public clouds, at the same time and for the same application.

There are many reasons for a company to use a private cloud. In some cases companies prefer to keep their application, and especially their data, locally for compliance reasons; in other cases this is due to security concerns or simply to avoid the burden of moving a lot of data; finally more and more companies choose to go back to a private or hybrid cloud solution after experiencing high costs when using the public one. For all those reasons we think that ADAPT should support hybrid cloud, thus combining the benefits of public clouds with those of private ones.

There are multiple hybrid scenarios that could be supported by ADAPT. The main difference among these scenarios is the location where ADAPT is executed, whether locally or in the cloud. Another thing that could change is whether the private cloud is accessed through ACSmI or directly; in the second case ADAPT would need to be extended to use different plugins both for deployment and for monitoring. ACSmI is already being extended to be able to provision resources in an OpenStack-based private cloud.

The hybrid scenario which is currently being implemented involves running ADAPT in the cloud and provisioning private cloud resources through ACSmI, as shown in Figure 15. The complexity in implementing this scenario comes from opening the private cloud provisioning API to the internet for ACSmI to access it.

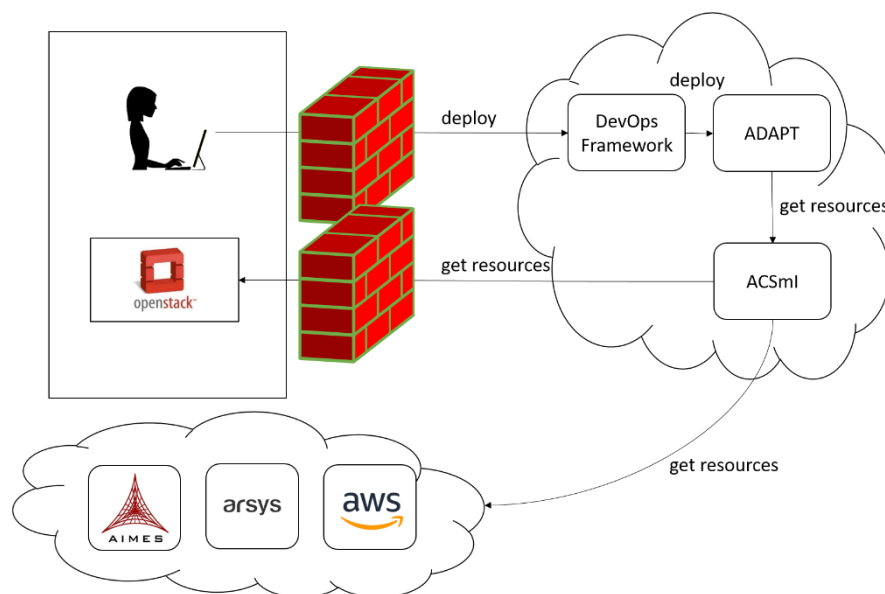


Figure 15. ADAPT Hybrid scenario

A further hybrid scenario which could be implemented in the next release involves running ADAPT locally and provisioning private cloud resources directly and public cloud resources through ACSmI, as shown in Figure 16.

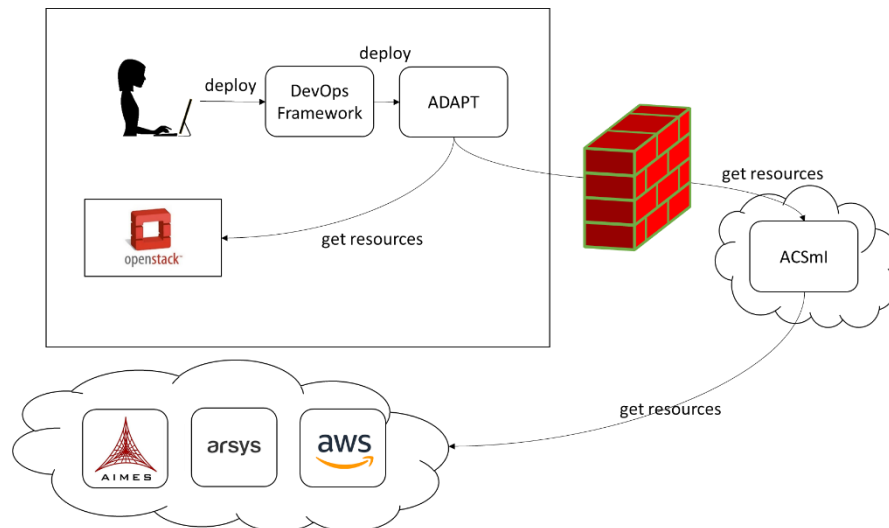


Figure 16. Further ADAPT Hybrid deployment scenario

### 6.3 ADAPT as a Service

There are several definitions of SaaS (Software as a Service). The SEI (Software Engineering Institute) defines it as a “Model of software deployment in which a third-party provider licenses an application to customers for use as a service on demand” [15].

When considering offering a SaaS, several architectural questions shall be answered to evaluate the implications and needs for this software model offering. In the following picture some of these questions are presented:

### SaaS: Examples of Architecture and Design Questions

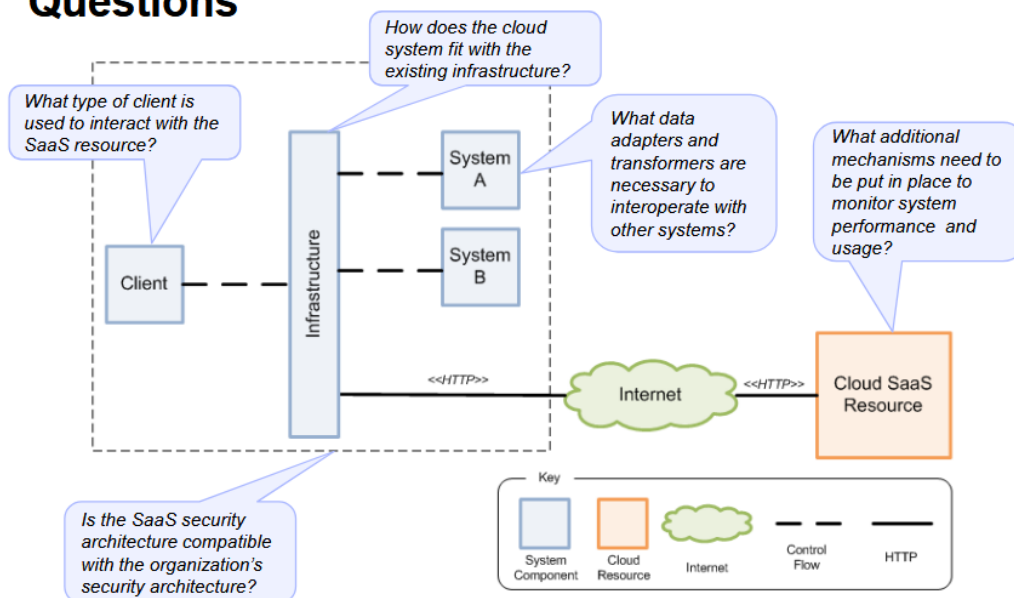


Figure 17. Examples of architectural and design questions in SaaS (source SEI [15])

These questions affect both the SaaS consumer and the SaaS providers. In the case of ADAPT we will analyze the implications of providing ADAPT as a Service, considering:

- Multi-tenancy requirements. Mainly in SaaS implementations, a tenant is an organization that makes use of the service. In this case a tenant should be an organization/user making use of ADAPT. Multi tenancy requires
  - Awareness of tenant context: the capability of recognizing the identity of the tenant requesting the resources based on message information as well as configuration data
  - Data isolation: tenants should only have access to their own data
  - Performance isolation: resource performance should conform to service
  - level agreements, regardless of the load on the system

Therefore, in the case of ADAPT multi-tenancy could be implemented in several ways:

- One instance per tenant: This will impact the sustainability of the model due to the need of resources.
- Single instance with tenant configuration data management: more efficient option but it implies to include a new component for managing the tenant configuration data.
- Multiple instance with load balancer: in this case the idea is to deploy identical instances managed by a load balancer. Here a configuration data management is also needed.
- Virtualization strategy. For the virtualization strategy several aspects need to be considered: How and when are virtual machines deployed, started, initialized, deactivated, replaced, managed and terminated?
- Resource interfaces. Configuration of interfaces for setting up the access to the application, view application usage data, etc. would be needed for offering ADAPT as a service.

## 7 Conclusions

This second year ADAPT architecture deliverable described the current version of DECIDE ADAPT's high-level architecture. The analysis of WP4 requirements has been improved and the number of requirements reduced from 51 to 17, by merging those dealing with the same functionality. ADAPT architecture has been described using UML syntax and the description includes the updated version of both internal and external interfaces, along with the current status of their integration. A discussion on scalability for the monitoring component has been included to highlight the benefits of selecting a Time Series Database for the implementation. ADAPT deployment scenarios have also been discussed, reporting about the architectural choice for a central deployment, describing the hybrid scenario which is currently being implemented, and discussing a possible ADAPT-as-a-Service scenario. In the next year further functionalities will be analysed, in addition to the planned support for manual readaptation cycle, such as: a further hybrid scenario, new monitoring metrics on redeployment, container replication, scalability and security for multi-tenant deployments, and possibly also container volumes.

## 8 References

- [1] DECIDE, "Deliverable D4.5 - Intermediate multi-cloud application deployment and adaptation".
- [2] DECIDE, "Deliverable D4.8 - Intermediate multi-cloud application monitoring".
- [3] DECIDE, "Deliverable D4.1 - Initial DECIDE ADAPT Architecture," 2017.
- [4] DECIDE, "Deliverable D6.2 - Final Use case Requirements Capture," 2018.
- [5] DECIDE, "Deliverable D2.5 - Detailed architecture v2".
- [6] DeHamer, "Gracefully stopping Docker containers," 2015. [Online]. Available: <https://www.ctl.io/developers/blog/post/gracefully-stopping-docker-containers/>. [Accessed Sept. 2018].
- [7] Burgess, "Why you should be using remote state in Terraform," 2017. [Online]. Available: <https://medium.com/@itsmattburgess/why-you-should-be-using-remote-state-in-terraform-2fe5d0f830e8>. [Accessed Sept. 2018].
- [8] InfluxData, "The Time Series Data Base explained," [Online]. Available: <https://www.influxdata.com/time-series-database/>.
- [9] "Time Series Database," Distributed Wikipedia, [Online]. Available: [https://ipfs.io/ipfs/QmXoypizjW3WknFIJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Time\\_series\\_database.html](https://ipfs.io/ipfs/QmXoypizjW3WknFIJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Time_series_database.html).
- [10] DB-engines, "DB-engines," [Online]. Available: <https://db-engines.com/en/>.
- [11] DB-engine initiative, "DBMS popularity broken down by database model," DB-initiative, [Online]. Available: [https://db-engines.com/en/ranking\\_categories](https://db-engines.com/en/ranking_categories).
- [12] "Time Series DB ranking," DB-engine, [Online]. Available: <https://db-engines.com/en/ranking/time+series+dbms>.
- [13] Basho, "Time Series Database explained," [Online]. Available: <http://basho.com/resources/time-series-databases/>.
- [14] S. N. Z. Naqvi, S. Yfantidou and E. Zimányi, "Time Series Databases and InfluxDB," 2017. [Online]. Available: [https://cs.ulb.ac.be/public/\\_media/teaching/influxdb\\_2017.pdf](https://cs.ulb.ac.be/public/_media/teaching/influxdb_2017.pdf).
- [15] G. Lewis, "Architectural Implications of Cloud Computing," Carnegie-Mellon Univ. Pittsburgh Pa Software Engineering Institute.
- [16] DECIDE-WP4-Partners, "DECIDE WP4 Requirements," [Online]. Available: [https://docs.google.com/spreadsheets/d/1ZopEEXmxXe\\_Rqr5xnZihh5bBjprXKJju0iwsAkfX1UE/edit#gid=0](https://docs.google.com/spreadsheets/d/1ZopEEXmxXe_Rqr5xnZihh5bBjprXKJju0iwsAkfX1UE/edit#gid=0). [Accessed 27 July 2017].
- [17] Wikipedia, "Time series Database," [Online]. Available: [https://en.wikipedia.org/wiki/Time\\_series\\_database](https://en.wikipedia.org/wiki/Time_series_database).

- [18] Time Scale, [Online]. Available: <https://blog.timescale.com/what-the-heck-is-time-series-data-and-why-do-i-need-a-time-series-database-dcf3b1b18563>.
- [19] G. A. Lewis, "Cloud Computing," Carnegie-Mellon Univ. Pittsburgh Pa Software Engineering Institute, 2009.