Open Data Center Alliance Master Usage Model: Service Orchestration Rev 2.0

TABLE OF CONTENTS

Legal Notice ..........................................................................................................................3
Dependency mapping ..............................................................................................................4
Executive Summary ..............................................................................................................5
Purpose ........................................................................................................................................6
Taxonomy.................................................................................................................................6
Introduction ..............................................................................................................................7
Reference Framework ...........................................................................................................8
Usage Scenarios ....................................................................................................................9
Usage Scenario 1 - Compose Service ....................................................................................9
Usage Scenario 2 - Submit Provisioning Request .................................................................10
Usage Scenario 3 - Reserve Resources for Service ...............................................................11
Usage Scenario 4 - Deploy Service .......................................................................................11
Usage Scenario 5 - Track Status and Manage Deployment ....................................................12
Usage Scenario 6 - Reopen Expired Request ......................................................................13
Usage Scenario 7 - Start Dependencies ...............................................................................14
Usage Scenario 8 - Stop Dependencies ................................................................................14
Usage Scenario 9 - Suspend Cloud Service .........................................................................15
Usage Scenario 10 - Resume Cloud Service ........................................................................15
Usage Scenario 11 - Systems Monitoring, alerting, and Data Collection .............................16
Usage Scenario 12 - Systems Administration and Remediation ............................................17
Usage Scenario 13 - Reporting ............................................................................................17
Usage Scenario 14 - Capacity Planning ..............................................................................18
Usage Scenario 15 - Auditing .............................................................................................19
Usage Scenario 16 - Change a Deployed Service Instance .....................................................20
Usage Scenario 17 - Autoscale within a Cloud .....................................................................21
Usage Scenario 17a - Bursting between Clouds ....................................................................21
Usage Scenario 18 - Comply to Regulatory Requirements .....................................................23
Usage Scenario 19 - Service and Data Termination and Deletion ...........................................24
Service Orchestration Architectural Overview .................................................................26
Machine-to-Machine Orchestration in the Context of API Communication and Orchestration Architecture ...............................................................28
Service Catalog Overview ..................................................................................................30
Services Interface Overview ...............................................................................................33
Key Performance Indicators Overview ...............................................................................36
Service Availability ...............................................................................................................37
Speed of Service Execution .................................................................................................38
Accuracy of Service Execution ............................................................................................38
Legal/Regulatory and Corporate Compliance of Service Output(s) .......................................39
Cost of Service Execution ....................................................................................................39
Cloud Subscriber Satisfaction ...............................................................................................40
Service Execution Incident Business Impact Ratio ............................................................40
Orchestration Lifecycle Overview .......................................................................................41
Initiation ...............................................................................................................................41
Run, Monitor, Change .........................................................................................................41
Termination ..........................................................................................................................41
Usage Requirements ............................................................................................................44
RFP Requirements – Service Providers ..............................................................................47
Summary of Industry Actions Required ..............................................................................48
Other Sources .....................................................................................................................48

CONTRIBUTORS

ODCA would like to acknowledge the substantial contributions of content and prior art from Atos, Bank Leumi, BMW, CapGemini, Deutsche Bank, Deutsche Telekom, NAB, UBS, and Wikipedia.

Avi Shwartz – Bank Leumi
Ian Lamont – BMW
Bernd Henning – Fujitsu
Mark Atwood – Hewlett-Packard
Anand Chaganty – Infosys
Chidambaram Ganapathi – Infosys
Sudip Chahal – Intel IT
Ravi A. Giri – Intel IT
Stephanie Woolson – Lockheed Martin
Charlie Tan – Microsoft Corporation
Mustan Bharmal – T-Systems
Ryan Skipp – T-Systems

TERMINOLOGY AND PROVENANCE

Some of the content of this document has been sourced from work produced outside the ODCA. Every effort has been made to reconcile terminology and nomenclature. Where there is a conflict, however, ODCA terms take precedence.
LEGAL NOTICE

© 2012-2013 Open Data Center Alliance, Inc. ALL RIGHTS RESERVED.

This “Open Data Center AllianceSM Usage Model: Service Orchestration Rev 2.0” document is proprietary to the Open Data Center Alliance (the “Alliance”) and/or its successors and assigns.

NOTICE TO USERS WHO ARE NOT OPEN DATA CENTER ALLIANCE PARTICIPANTS: Non-Alliance Participants are only granted the right to review, and make reference to or cite this document. Any such references or citations to this document must give the Alliance full attribution and must acknowledge the Alliance’s copyright in this document. The proper copyright notice is as follows: “© 2012-2013 Open Data Center Alliance, Inc. ALL RIGHTS RESERVED.” Such users are not permitted to revise, alter, modify, make any derivatives of, or otherwise amend this document in any way without the prior express written permission of the Alliance.

NOTICE TO USERS WHO ARE OPEN DATA CENTER ALLIANCE PARTICIPANTS: Use of this document by Alliance Participants is subject to the Alliance’s bylaws and its other policies and procedures.

NOTICE TO USERS GENERALLY: Users of this document should not reference any initial or recommended methodology, metric, requirements, criteria, or other content that may be contained in this document or in any other document distributed by the Alliance (“Initial Models”) in any way that implies the user and/or its products or services are in compliance with, or have undergone any testing or certification to demonstrate compliance with, any of these Initial Models.

The contents of this document are intended for informational purposes only. Any proposals, recommendations or other content contained in this document, including, without limitation, the scope or content of any methodology, metric, requirements, or other criteria disclosed in this document (collectively, “Criteria”), does not constitute an endorsement or recommendation by Alliance of such Criteria and does not mean that the Alliance will in the future develop any certification or compliance or testing programs to verify any future implementation or compliance with any of the Criteria.

LEGAL DISCLAIMER: THIS DOCUMENT AND THE INFORMATION CONTAINED HEREIN IS PROVIDED ON AN “AS IS” BASIS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THE ALLIANCE (ALONG WITH THE CONTRIBUTORS TO THIS DOCUMENT) HEREBY DISCLAIM ALL REPRESENTATIONS, WARRANTIES AND/OR COVENANTS, EITHER EXPRESS OR IMPLIED, STATUTORY OR AT COMMON LAW, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, VALIDITY, AND/OR NONINFRINGEMENT. THE INFORMATION CONTAINED IN THIS DOCUMENT IS FOR INFORMATIONAL PURPOSES ONLY AND THE ALLIANCE MAKES NO REPRESENTATIONS, WARRANTIES AND/OR COVENANTS AS TO THE RESULTS THAT MAY BE OBTAINED FROM THE USE OF, OR RELIANCE ON, ANY INFORMATION SET FORTH IN THIS DOCUMENT, OR AS TO THE ACCURACY OR RELIABILITY OF SUCH INFORMATION. EXCEPT AS OTHERWISE EXPRESSLY SET FORTH HEREIN, NOTHING CONTAINED IN THIS DOCUMENT SHALL BE DEEMED AS GRANTING YOU ANY KIND OF LICENSE IN THE DOCUMENT, OR ANY OF ITS CONTENTS, EITHER EXPRESSLY OR IMPLICIEDLY, OR TO ANY INTELLECTUAL PROPERTY OWNED OR CONTROLLED BY THE ALLIANCE, INCLUDING, WITHOUT LIMITATION, ANY TRADEMARKS OF THE ALLIANCE.

TRADEMARKS: OPEN CENTER DATA ALLIANCESM, ODCA℠, and the OPEN DATA CENTER ALLIANCE logo® are trade names, trademarks, and/or service marks (collectively “Marks”) owned by Open Data Center Alliance, Inc. and all rights are reserved therein. Unauthorized use is strictly prohibited. This document does not grant any user of this document any rights to use any of the ODCA’s Marks. All other service marks, trademarks and trade names reference herein are those of their respective owners.

© 2012-2013 Open Data Center Alliance, Inc. ALL RIGHTS RESERVED.
DEPENDENCY MAPPING

Below is the dependency mapping between this document and other publications. The list of documents above the blue arrow show which ODCA documents feed into Service Orchestration. The list of documents below the blue arrow indicate other critical inputs from industry books, standards, and references.
EXECUTIVE SUMMARY

Service offerings from cloud providers are becoming more complex and open, and it will become increasingly important to ensure that cloud infrastructure requirements are aligned with cloud consumer demand and can scale as needed. This includes providing the underlying physical resources, such as servers, networks, storage, and hosting infrastructure, and the cloud software that renders these resources in the form of platform as a service (PaaS), software as a service (SaaS), and other categories. Additionally, the concept of mobility of services and interoperability are expected, with supporting automation. In this context, well-defined service orchestration becomes essential to developing and providing cloud services that are relevant to a variety of cloud role players, including consumers, providers, brokers, and agents.

Service orchestration is a paradigm that supports cloud providers in arranging, coordinating, and managing computing resources as a system of components and automated workflows that can be delivered as cloud services to cloud consumers. The scope of this involves three types of system components—the service layer, the resource abstraction and control layer, and the physical resource layer—which are the underlying foundations of the cloud provider’s offerings provided in a service catalog.

This document, “ODCA Master Usage Model: Service Orchestration,” illustrates the usage model for service discovery and orchestration for cloud services. This usage model includes the components, repositories, data structures, conversation maps, and workflows that assist a cloud consumer in defining required services, and the associated cloud provider to orchestrate services to meet objectives determined by the cloud subscriber and for the appropriate context of use. It defines automation required for orchestration that includes programmatic interfaces, interaction patterns, control interfaces, and lifecycle management.

This usage model includes 19 usage scenarios describing various stages of a cloud service engagement, including ordering, starting and stopping, running, changing, and ending or deleting cloud services. It also lays out the foundation for the next phase of usage development, such as bursting between clouds and other areas of the cloud that include PaaS, SaaS, and so on.

Finally, the document makes recommendations for industry actions to further develop capabilities that are foundational to building a marketplace of services that allows a cloud consumer to source well-defined and standards-based services.

While services composed in such orchestrations can be sourced from either public, private, community, or hybrid clouds, the focus of this usage model is the cloud subscriber sourcing services from public cloud providers. The orchestration could be as simple as placing constraints on the use of the service to more complex orchestrations that are a composition of more elemental services to derive an overall higher level of service.

This usage model serves a variety of audiences. Solution providers and technology vendors will benefit from its content to better understand customer needs, and to their tailor service and product offerings. Standards organizations will find the information helpful in defining end-user relevant open standards.
PURPOSE
This document focuses on service discovery and orchestration for cloud services. It defines the automation required for orchestration that includes programmatic interfaces, interaction patterns, control interfaces, and lifecycle management.

The goals of this usage model are as follows:

• Define the basic elements of service orchestration for cloud services.
• Apply the baseline work on the ODCA Usage Model: Service Catalog¹ and ODCA Usage Model: Standard Units of Measure for IaaS.²
• Identify the ODCA Usage Model: Service Catalog, API interface, key performance indicators (KPIs), and process for service orchestration.
• Align with the ODCA Master Usage Models: Commercial Framework³ and Compute IaaS (IaaS)⁴ to underpin cloud services provisioning.
• Define a standard orchestration process for service integration that can be used as a reference model for improving interoperability between cloud providers and subscribers.

TAXONOMY
Table 1 lists the standard terms and definitions used in this document.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autoscaling</td>
<td>The ability to automatically provision a new service instance or remove an existing service instance in a collection as defined by a set of one or more policies. Autoscaling also includes the ability to alter a configuration to align more directly on usage requirements.</td>
</tr>
<tr>
<td>Bursting</td>
<td>The ability for a cloud consumer to scale out its infrastructure from one cloud provider into another, typically from a private cloud provider to a public cloud provider.</td>
</tr>
<tr>
<td>Collection</td>
<td>Describes a set of one or more service instances.</td>
</tr>
<tr>
<td>Cloud Broker</td>
<td>A type of software that sits on top of cloud providers to abstract, simplify, and map various cloud offerings to defined requirements. Cloud broker software assists organizations in creating solutions in the cloud, migrating solutions to the cloud, and moving solutions between clouds.</td>
</tr>
<tr>
<td>Migration</td>
<td>The ability for cloud consumer to move a service instance from one cloud provider to another.</td>
</tr>
<tr>
<td>Post Deployment Task</td>
<td>Represents a task to be executed after the deployment of service instance, typically in the form of a script that the cloud consumer provides.</td>
</tr>
<tr>
<td>Reservation</td>
<td>A confirmation from the cloud provider that the specific set of resources specified in a service request has been reserved for the cloud consumer.</td>
</tr>
<tr>
<td>Service Composition</td>
<td>The action of defining the specification for a set of interrelated components that make up a service.</td>
</tr>
<tr>
<td>Service Instance</td>
<td>An occurrence or instantiation of the service as listed in the catalog that is deployed into the environment for a cloud consumer.</td>
</tr>
<tr>
<td>Service Orchestration</td>
<td>The ongoing ability to arrange, coordinate, and manage the automated deployment and configuration of one or more interrelated components required for service delivery at a point in time.</td>
</tr>
<tr>
<td>Service Request</td>
<td>An order submitted by the cloud consumer that details the service and its composition that the cloud provider is expected to deliver.</td>
</tr>
</tbody>
</table>

¹ See www.opendatacenteralliance.org/library
² See www.opendatacenteralliance.org/library
³ See www.opendatacenteralliance.org/library
⁴ See www.opendatacenteralliance.org/library
INTRODUCTION

Service discovery and orchestration is a “paradigm that supports cloud providers in arranging, coordinating, and managing computing resources as a system of components and automated workflows that can be delivered as cloud services to cloud consumers.”

At the most basic level a service orchestrator can be a human, but for the purpose of this document, an orchestrator is an automation service that provides orchestration across various technical and business workflow domains. Because the cloud is all about scale, automated workflows are essential to the delivery of cloud services, which include fulfillment assurance and billing, in addition to other domains.

The main difference between workflow automation and service orchestration is that automated workflows represent the entities and execution relationship; automation is the non-human service that drives the workflow. These workflows are often processed and completed as processes with a single domain. Service orchestration spans multiple domains and layers, including technology, commercial, and service layers.

On the other hand, service orchestration includes a workflow, but implies directed action toward larger goals and objectives. In a nutshell, it is different from a typical workflow automation process because it ties together a variety of different or “disparate automated processes and IT resources that use workflows through a portal from which those workflows can be managed.”

Service orchestration for any type of cloud service involves specific considerations. Functional, non-functional, and constraint descriptions must be clearly defined. Introspection of a service (and the consumers’ service requirements) may be useful in determining these details. A well-defined service catalog where these services can be looked up to determine which services are available to meet required functionality as well as the interfaces that support them is a must-have.

Discovery is a process of assessing the capabilities of the services and contracts as well as the commercial parameters that allow efficient transactions that are highly secure with great elasticity. In the future, with established marketplaces for services, it will be possible to have pre-qualified service providers and services, and discovery time should be reduced. Certain filters may be applied to the discovered results; for example, if the consumer corporation has defined carbon efficiency criteria, then only those discovered capabilities that fall within the corridor of carbon requirements will be parsed. Similar additional criteria may include geography, compliance and legislative parameters, and scalability potential (for example, storage capability to scale out). See the ODCA Master Usage Model: Commercial Framework for further drivers. A number of data security factors (See ODCA Usage Model: Data Security) must also form part of the pre-filtering metadata, in order to be able to match requirements and possible targets. These factors include defined parameters for the following:

- Access control
- Information classification
- Data encryption
- Data masking
- Security information and event management
- Backup, archiving, and deletion

Service reservation or a declaration of intent and when orders will be filled, such as when the service and resources are filled, are important for fulfillment of a service.

Identification of the methods for service orchestration of enhanced services, customization, or mashups is an important part of orchestration. This also includes requirements related to business processes, outsourced segments of the business process to external cloud services, and orchestration for managing contracted services. Defining the order of services may be necessary to identify other dependencies, such as lifecycle management, interoperability of environments, bootstrapping, and so on.

Security considerations are a necessary requirement to delivering a robust cloud service. For example, authentication, authorization, audit, and accounting (AAAA) are essential to service orchestration as consumers evaluate topics such as federated identity. In addition, access to the service orchestration capability itself (including interfaces, functionality, and processes) must be managed according to strict governance and security controls.

Service-level agreements (SLAs) are also essential to the fulfillment of cloud services. This might take place either in-band, out-of-band, or in advance of a specific engagement.

---

6 See www.opendatacenteralliance.org/library
7 See www.opendatacenteralliance.org/library
In order to use a common language for defining requirements and evaluating options, it is useful for both cloud subscribers and cloud providers to use common metrics and common definitions for elements. The ODCA Usage Model: Standard Units of Measure\(^8\) provides a useful foundational set of conventions, for both groups to consider.

Leveraging existing usage models from the ODCA workgroup repositories, this paper focuses on how service components are combined based on the IaaS framework. It generally takes the viewpoint of the service consumer but also addresses the viewpoints of the service providers and vendors.

This usage model contains 19 service orchestration usage scenarios between a cloud subscriber and a public cloud provider IaaS. It also lays the foundation for the next phase of usage development in other areas of the cloud such as “X”aaS. It identifies what needs to be configured for the IaaS cloud service, what must appear in the service catalog, as well as how the consumer orders services, changes, and decommissioning, and how it is proved that they are receiving the service.

**REFERENCE FRAMEWORK**

Figure 1 illustrates five high-level groupings of usage scenarios based on the orchestration tasks that cloud subscribers execute, which also map closely to the industry concepts of Install, Move, Add, Change, and Delete.

More detail about each usage scenario is provided in the sections that follow.

---

**Cloud Service Lifecycle**

Orchestration Usage Scenario

![Cloud Service Lifecycle Diagram](image)

1. Compose Service
2. Submit Provisioning Request
3. Reserve Resources for Service
4. Deploy Service
5. Track Status and Manage Deployment
6. Reopen Expired Request
7. Start Dependencies
8. Stop Dependencies
9. Suspend Cloud Service
10. Resume Cloud Service
11. System Monitoring, Alerting, and Data Collection
12. System Administration and Remediation
13. Reporting
14. Capacity Planning
15. Auditing
16. Change a Deployed Service Instance
17. Autoscale within a Cloud
18. Comply to Regulatory Requirements
19. Service and Data Termination and Deletion

Figure 1. High-level grouping of service scenarios.

---

\(^8\) See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)
**Usage Scenarios**

General note: Every usage scenario can be implemented either via a graphical user interface or programmatically via application programming interfaces (APIs).

**Usage Scenario 1 - Compose Service**

**Actors**  
Cloud subscriber, cloud provider

**Goal**  
A composition service order with advanced structures and dependencies, and the ability to manage deployment across multiple cloud providers.

**Assumption 1**  
An order for a complex service composition can target deployment to more than one service provider.

**Assumption 2**  
Cloud subscriber should expect data validation, resource reservation, and release for deployment, and the progress tracking should be functionally similar.

**Success Scenario 1**  
The ability of the cloud subscriber to compose a simple or more complex service request and structure. The simplest form is an order that requests only a single service type (that is, a virtual machine (VM) instance). The complex composition proposed here is to allow the cloud subscriber to nest service requests within a single order. An example is a request that is composed of multiple VM instances with post-deployment tasks defined so that successful fulfillment of the order will result in an application that is ready to run. There are more advanced customizations that must be considered, such as but not limited to priority/dependency on the order of deployment. Service orchestration is visibly more relevant here and must be able to reliably determine when an instance is successfully deployed so that the orchestrator can move forward with the next deployment/s in the priority/dependency list.

**Steps**

1. Cloud subscriber creates a new composition for a service: For each service instance that the cloud subscriber wants to embed into this new request, the cloud subscriber repeats the following:
   - Select the cloud provider.
   - Define the service identifier.
   - Define the identity of the service process (technical user identity).
   - Define the cloud subscriber’s operation support.
   - Select predefined user group/role that will have rights to manage the instance of this service based on the ODCA Usage Model: Identity Management Interoperability Guide.9
   - Select the appropriate ODCA Usage Model: Security Monitoring10 requirements.
   - Select the security requirements based on the Bronze, Silver, Gold, and Platinum tiers defined in the ODCA Usage Model: Provider Assurance.11
   - Determine the service instance identifier (instance’s identifier) (that is, the network address of the server).
   - Define the service’s runtime attributes such as memory, vCPU, storage capacity, latency, IOPS and latency, network bandwidth and quality-of-service (QoS) requirements, and location.
   - Choose the desired commercial and financial terms, SLA, qualities of service and availability for delivery of service.
   - Select the required post-deployment tasks and automation scripts.

2. Cloud subscriber identifies dependencies in deployment (such as when instance #2 cannot start until instance #1 is successfully completed).

3. Cloud subscriber selects notification methods, such as email or text.

---

9 See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)  
10 See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)  
11 See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)
Failure Condition 1  
Data is incomplete.

Failure Handling 1  
The subscriber is notified and prompted to correct the composition.

Failure Condition 2  
Cyclic dependencies are identified and erred.

Failure Handling 2  
The subscriber is notified and prompted to correct the composition.

Usage Scenario 2 - Submit Provisioning Request

Actors  
Cloud subscriber, cloud provider

Goal  
Validate, reserve resources (trigger Usage Scenario 3 - Reserve Resources for Service) and submit service request.

Assumption 1  
Usage Scenario 1 - Compose Service has been completed.

Assumption 2  
A new order process has been started but not yet released for deployment.

• If data is complete and resources reserved, the cloud subscriber can release the order.
• If resource reservation has expired, the cloud subscriber can submit the request again.
• If data is complete but one or more resources are not available, the cloud subscriber can make adjustments or request the cloud provider to reserve the resources for the order when they become available.
• If data is not complete, the order is saved as incomplete, and the cloud subscriber can continue at a later date and time.
• At any given time, the cloud subscriber can delete the order, and any resources already reserved will be released back into the resource pool.

Success Scenario 1  
Cloud subscriber submits the request to provision a service. On submission, cloud provider checks for resource availability, and reserves resources. Cloud subscriber reviews and approves the final order prior to release for automatic deployment.

Steps

1. Cloud subscriber selects the composed service created in Usage Scenario 1 - Compose Service and submits to the cloud provider/s.
2. On submission of the request to the cloud provider:
   – Cyclic dependency is identified. If a cyclic dependency is identified by the cloud provider, the dependency is highlighted and prompts the cloud subscriber to review.
   – Each cloud provider checks to see that all requested resources in the order are available and meet the specifications.
   – If any request data is incomplete, the missing data is highlighted, and prompts the cloud subscriber to review.
   – If the order can be met, the cloud provider reserves the resources for the cloud subscriber (trigger Usage Scenario 3 - Reserve Resources for Service). The reservation has an expiration date.
   – If the order cannot be met, the cloud provider will provide an estimated date in the future when the cloud subscriber can expect resources to become available.
3. Cloud subscriber reviews the order prior to release for automatic deployment.

Failure Condition 1  
Request data is incomplete.

Failure Handling 1  
The subscriber is notified and prompted to complete the request.
Failure Condition 2
Cloud provider does not have capacity to reserve resources for the cloud subscriber.

Failure Handling 2
The subscriber is notified and has the option of cancelling the order or waiting until resources are available.

**Usage Scenario 3 - Reserve Resources for Service**

**Actors**
Cloud subscriber, cloud provider

**Goal**
Reserve resources for service request.

**Assumption 1**
*Usage Scenario 2 - Submit Provisioning Request* has started.

**Assumption 2**
No pre-existing orders exist with any resources that have the same identifier in the namespace. Any organization level resources, such as user groups, networks, and firewalls, are predefined and selectable.

**Success Scenario 1**
Cloud provider checks for resource availability and reserves resources.

**Steps**
Cloud provider reserves the resources for the cloud subscriber. The reservation has an expiration date.

**Failure Condition 1**
Cloud provider does not have capacity to reserve resources for the cloud subscriber.

**Failure Handling 1**
The subscriber is notified and can resubmit the order at a later time. Ideally the provider notifies the subscriber when capacity is available.

**Usage Scenario 4 - Deploy Service**

**Actors**
Cloud subscriber, cloud provider

**Goal**
Deploy service request.

**Assumption 1**
Cloud provider confirmed resource availability and resource reservation has not expired.

**Assumption 2**
The service order has been released for automatic deployment. The cloud subscriber cannot make any more changes to the service order. The cloud subscriber has an option to cancel the deployment.

**Success Scenario 1**
The cloud subscriber releases the service request for automatic deployment.

**Steps**
1. Cloud subscriber selects from a list of open service requests.
2. Cloud subscriber releases the service request for automatic deployment.

**Failure Condition 1**
Resource reservation has expired.

**Failure Handling 1**
The subscriber is notified and has to reopen the service request or start a new one.
Failure Condition 2
Cloud subscriber cancels the order and any resources reserved are released back into the resource pool.

Failure Handling 2
Any resources reserved are released back into the resource pool.

Failure Condition 3
Cloud subscriber cancels the deployment only. The service request is still open and resources are still reserved.

Failure Handling 3
The service request is still open and resources are still reserved, awaiting the cloud subscriber’s next action or until the request expires.

Usage Scenario 5 - Track Status and Manage Deployment

Actors
Cloud subscriber, cloud provider

Goal
Track progress of a service request and manage state. The cloud subscriber wants to manage a service request that is currently being deployed.

Assumption 1
The cloud subscriber already submitted the service request for automatic deployment, and it is currently in progress.

Assumption 2
Deployment status is transparent to the cloud subscriber.

Success Scenario 1
After a service request has been released for deployment, the cloud subscriber can track progress of the deployment and has some ability to see the state and manage some aspects of it.

Steps
1. The cloud subscriber selects the service request.
2. The cloud provider returns a summary about the request. The summary should include but not be limited to the following information: request ID, name of request, type of service and cloud provider, list of resources specified in the request, deployment status.
   The status report should include but not be limited to the following list of information: Percentage to completion, estimated time to completion, status of different resources in the request, error messages, if any, and activity log.
3. Depending on the current state of deployment, the cloud subscriber may choose one of the following actions:
   - Cancel the deployment, roll back any changes, and put resources back into the pool.
   - Pause the deployment at the next checkpoint.
   - Resume a deployment that was paused.
   - Fix errors and retry/resume deployment from the last checkpoint.

Failure Condition 1
Provisioning request is not progressing, and there is no error message.

Failure Handling 1
Subscriber has a self-service means to check progress and escalate within the provider’s support organization.

Failure Condition 2
An error occurred, and the deployment was aborted by the cloud provider.

Failure Handling 2
Subscriber is notified and can either correct and resubmit, or cancel the request.
Usage Scenario 6 - Reopen Expired Request

Actors
Cloud subscriber, cloud provider

Goal
Reuse an expired service request without having to start from scratch.

Assumption 1
The cloud subscriber previously submitted a service request but the resource reservation has (already) expired (for example, a fixed-period system request, prepaid service, or just an incomplete workflow process).

Assumption 2
A new order process has been started but not yet released for deployment.
• If the request data is complete and resources are reserved, the cloud subscriber can release the order.
• If the resource reservation has expired, the cloud subscriber can submit the request again.
• If the request data is complete but one or more resources are not available, the cloud subscriber can make adjustments or the cloud subscriber can request the cloud provider to reserve the resources for the order when they become available.
• If request data is not complete, the order is saved as incomplete, and the cloud subscriber can continue at a later date and time.
• At any given time, the cloud subscriber can delete the order, and any resources reserved will be released back into the resource pool.

Success Scenario 1
The cloud subscriber submitted the request for service previously. Resources were reserved by the cloud provider and have already expired. The cloud subscriber can edit the request and resubmit.

Steps
1. The cloud subscriber selects from a list of expired service requests.
2. The cloud subscriber updates the service request. For example: service type, reference number, resources, dependencies, and customizations.
3. On submit, the cloud provider verifies the completeness of the request data.
4. If the request data is incomplete, the cloud provider highlights the missing information and prompts the cloud subscriber to review and update the request with the missing information.
5. On successful submission, the cloud provider checks to see if the requested resources in the order can be met.
6. If the order can be met, the cloud provider reserves the resources for the cloud subscriber (triggers Usage Scenario 3 - Reserve Resources for Service). Each reservation has an expiration date.
7. If the order cannot be met, the cloud provider gives an estimated date in the future when the cloud subscriber can expect resources to become available.
8. The cloud subscriber reviews the order one last time, prior to release for automatic deployment.

Failure Condition 1
The request data is incomplete.

Failure Handling 1
The subscriber is notified and prompted to complete the request.

Failure Condition 2
The cloud provider does not have the capacity to reserve resources for the cloud subscriber.

Failure Handling 2
The subscriber is notified and can resubmit the order at a later time. Ideally the provider notifies the subscriber when capacity is available.
Usage Scenario 7 - Start Dependencies

Actors
Cloud subscriber, cloud provider

Goal
After successful reservation and initialization, cloud service is started.

Assumption 1
The cloud service provider confirmed resource availability.

Assumption 2
The starting date and time do not exceed reservation period for that specific request for all resources.

Assumption 3
The cloud provider has all necessary and relevant request data for the specific request.

Success Scenario 1
Cloud service is provided and started at the right point in time and at the correct quality level according to the cloud subscriber’s request.

Steps
1. Start of cloud service (ensure right starting order of dependent resources, if applicable).
2. Execution of cloud subscriber defined test scenario for successful service execution; measurement of its runtime if applicable (optional).
3. Return instance ID, starting times, status, and runtime to cloud subscriber.
4. Generate unique billing record.

Failure Condition 1
Cloud service not started or execution of test scenario was unsuccessful.

Failure Handling 1
Escalation through cloud provider’s support path.

Usage Scenario 8 - Stop Dependencies

Actors
Cloud subscriber, cloud provider

Goal
Cloud service is stopped, reserved resources disengaged.

Assumption 1
Cloud service is running.

Assumption 2
Requestor is entitled to stop the service.

Success Scenario 1
Cloud service is stopped, resources are freed up. ID and configuration data are kept for billing and future usage.

Steps
1. Cloud service is stopped. The correct order of stopping dependencies is ensured.
2. Resources used by cloud service that can be freed up are released and made available to the resource pool again.
3. ID, configuration, and billing record are kept for future re-initialization (see Usage Scenario 7 - Start Dependencies).

Failure Condition 1
Cloud service is not stopped at the designated time.

Failure Handling 1
Tier 2 support of cloud provider.
Failure Condition 2
Storage capacity for data is released to the available pool when the system is stopped, and data is lost.

Failure Handling 2
The cloud subscriber updates the data handling portion of the contract and the process for handling the scenario of a machine being stopped for a defined period.

Usage Scenario 9 - Suspend Cloud Service

Actors
Cloud subscriber, cloud provider

Goal
Cloud service is suspended.

Assumption 1
Cloud service is running at cloud provider.

Assumption 2
It is ensured that the requestor is entitled to suspend the service.

Success Scenario 1
Cloud service with all associated resources is suspended; resources that are no longer needed by the service are released.

Steps
1. Cloud service is suspended, checkpoint data is written, if applicable.
2. Resources stay reserved for the suspended service except those resources, such as compute, that are no longer required in the suspended state.
3. Billing record for unique ID is updated.

Failure Condition 1
Cloud service is not suspended.

Failure Handling 1
Tier 2 support of cloud provider.

Failure Condition 2
Storage capacity for data is released to the available pool when the system is suspended, and data is lost.

Failure Handling 2
Cloud subscriber updates the data handling portion of the contract and the process for handling the scenario of a machine being stopped for a defined period.

Usage Scenario 10 - Resume Cloud Service

Actors
Cloud subscriber, cloud provider

Goal
Cloud service is resumed.

Assumption 1
Cloud service is suspended before this stage.

Assumption 2
It is ensured that requestor is entitled to resume the service.

Success Scenario 1
Cloud service with all associated resources is resumed. If applicable, the service resumes from the last saved checkpoint status.
Steps
Cloud service is resumed. The right order of resuming resources is ensured.

Failure Condition 1
Cloud service is not resumed, not resumed at the designated time, or just partially resumed.

Failure Handling 1
Tier 2 support of cloud provider.

Usage Scenario 11 - Systems Monitoring, Alerting, and Data Collection

Actors
Cloud subscriber, cloud provider

Goal
Timely notification of urgent and important activity within the service, including health, performance, and billing aspects.

Assumption 1
The service has a set of KPIs that are configured based on published SLA and a contractual agreement between the subscriber and provider.

Assumption 2
Defined monitoring points for data collection measure both KPI input requirements and compliance.

Assumption 3
Monitoring is based on proactive, real-time architecture for all relevant KPI elements, according to a process that is compliant with applicable country and corporate governance as well as subscriber-specified smoothing calculations (to filter out noise from transient conditions).

Assumption 4
A database has been created to store trend data, including ongoing maintenance.

Success Scenario 1
The cloud subscriber receives an alert message when an established threshold is triggered.

Steps
1. Thresholds and associated messaging are configured.
2. The system detects that an established threshold is reached in the running system, such as health, performance, or billing conditions.
3. The trigger action is automatically executed to send an alert message in the proper formats (email, SMS, and message queue).
4. Identified data for trend analysis or reporting is posted to a database.
5. Data is retained for a sufficient period to enable analysis and compliance.

Failure Condition 1
Trigger condition is realized, but alert is not generated.

Failure Handling 1
Error in alerting is logged and escalated. The cloud provider delivers a manual alert.

Failure Condition 2
The correct data is not retained in a database to enable sufficient reporting or trend analysis.

Failure Handling 2
Update monitors to collect the correct data and to retain it in a suitable location.

Failure Condition 3
An agreed threshold and or smoothing/filtering algorithm was not established/provided/captured by the cloud provider system.

Failure Handling 3
Technical business remediation, according to the agreement.
Usage Scenario 12 - Systems Administration and Remediation

Actors
Cloud subscriber, cloud provider

Goal
Take a prescribed action in response to an alert to correct a specific defect or deficiency detected within the cloud service; for example, an autoscale action in response to increasing load on the service.

Assumption 1
The cloud service solution provides a mechanism to execute a remediation routine or script.

Assumption 2
Events are defined for all cloud service elements, with associated mapping to remediation routines or scripts. Step-out and escalation conditions exist and are automated.

Success Scenario 1
Successful remediation executed.

Steps
1. All events and remediation routines are defined.
2. A defined event is detected in the running system.
3. Each action associated with the event is executed.
4. Remediation of the condition causing the event is verified; if the condition has not been remediated, a FATAL alert is raised and the service is stopped.

Failure Condition 1
Failure to trigger a remediation routine or script.

Failure Handling 1
Identify the trigger and correct remediation routine or script, and add them to the automation mechanism.

Failure Condition 2
Remediation routine or script fails.

Failure Handling 2
Update the routine or script if applicable, or correct the preconditions expected or required by the routine or script.

Usage Scenario 13 - Reporting

Actors
Cloud subscriber, cloud provider

Goal
Produce reports according to the defined KPIs and publish them to the authorized parties to enable proactive management and planning.

Assumption 1
Defined reports form a part of the service and align to the defined KPIs for the service.

Assumption 2
Ad hoc reports can be generated as required.

Assumption 3
The monitoring system collects and retains sufficient data relating to the KPIs to enable reporting, in both real time and historical dimensions.

Assumption 4
All reports are authenticated and tamper resistant, such as integrity checks and time stamps.

Success Scenario 1
All defined reports are rendered accurately and represent the real system.
Steps
1. Reports are defined to show evidence that the expected service is indeed being delivered according to KPIs. These include availability, performance, capacity, service level achievement, compliance, commercial, security and other reports that may be deemed necessary from time to time.
2. Data is mined automatically to create reports according to a defined timing.
3. Reports are published automatically according to a defined timeline.
4. Based on available data, ad hoc reports can be created by the cloud subscriber.

Failure Condition 1
Reports are not defined.

Failure Handling 1
Identify the relevant reports that must be produced and the KPI-related data to support them, and produce the report.

Failure Condition 2
Reports are inaccurate.

Failure Handling 2
Identify if incorrect data is being used or if inadequate data is collected, and correct the gap.

Failure Condition 3
Report data is not retained.

Failure Handling 3
Identify data and report retention requirements. Correct for future report generation.

Failure Condition 4
Reports are inadequate to determine key service factors.

Failure Handling 4
Identify the correct source data needed to represent the service factor, collect it, and update the report.

Failure Condition 5
Compliance requirements are not met.

Failure Handling 5
Review the process required to collect the data and produce the report and identify what gaps exist to achieve compliance, and correct the process.

Usage Scenario 14 - Capacity Planning

Actors
Cloud subscriber, cloud provider

Goal
Produce adequate data automatically for all parties to be able to predict service dimensions, capital expenditure, operational costs and potential problems and incidents, and to be able to plan capacity requirements appropriately.

Assumption 1
Correct data to support capacity planning is collected and retained for an adequate period.

Success Scenario 1
The capacity planning process mines the correct data and automatically provides timely planning inputs that enable sustainable services and thereby avoids incident and problem occurrence.
Steps
1. All capacity reporting dimensions are defined.
2. Supporting data collection is scheduled or automated.
3. Capacity data is analyzed automatically from the available source.
4. Forecasts of events and changes are automatically produced.
5. The information is sent to the cloud subscriber for approval, if required by the contract.

Failure Condition 1
Service quality is impacted.

Failure Handling 1
Identify the relevant data to analyze and develop a trend line that includes defined thresholds to trigger proactive forecasting of any necessary decision or change points.

Failure Condition 2
Unplanned costs arise.

Failure Handling 2
Identify the relevant data to analyze and develop a trend line with thresholds to trigger forecasting of decision/change points.

Usage Scenario 15 - Auditing

Actors
Cloud subscriber, cloud provider, third-party auditor

Goal
Produce evidence that the service is produced according to both country-compliant and corporate-compliant processes as well as within applicable legal boundaries.

Assumption 1
All needed data is identified, collected, and retained according to processes compliant with country and corporate requirements.

Assumption 2
An auditor is used who correctly interprets both corporate and country requirements.

Assumption 3
Some auditing is automated, and some audits are manual and ad hoc. Data exists to support both methods.

Assumption 4
All reports are authenticated and tamper resistant, including integrity checksums and time stamps.

Success Scenario 1
All services are proved to be rendered according to expectations (defined by agreed-upon KPIs) and in a manner that meets the compliancy requirements of both the corporation and the country.

Steps
1. Review whether the necessary data is available to support the reporting on the agreed-upon or identified KPIs.
2. Review whether the method used to collect the data meets compliancy requirements.
3. Review whether the method used to convert the data into reports meets compliancy requirements.
4. Schedule the audit activities and reports that can be automated.
5. Execute the defined auditing activities per the schedule and in response to ad hoc requests.

Failure Condition 1
The required data to support KPI reporting is not available.

Failure Handling 1
Identify the needed data, its source, and any other factors, such as retention requirements, and update the monitoring system to collect it.
Failure Condition 2
Unacceptable methods are used to collect the data.

Failure Handling 2
Identify the correct method to be used for collecting the relevant data, and correct the method or process.

Failure Condition 3
Unacceptable methods are used to convert the data into reports.

Failure Handling 3
Identify the correct method for interpreting the data into compliant reports, and then correct the process.

Usage Scenario 16 - Change a Deployed Service Instance

Actors
Cloud subscriber, cloud provider

Goal
Process defined for changes, change approvals, provisioning, and tracking of the entire cloud service lifecycle.

Assumption 1
All scalability and change options are based on items defined in the service catalog. No changes deviate from options defined in the catalog.

Assumption 2
Infrastructure has fault tolerant, load balanced cluster and auto failover of compute, network, and storage (as configured).

Assumption 3
A set of rules exist defining what scalability options exist that can be accommodated for both changes and scaling workloads.

Success Scenario 1
Enhances the cloud consumer experience.

• Gives cloud consumers a unified experience and dramatically simplifies requests for cloud resources.
• Removes demand bottlenecks while enforcing approval policies, through service automation.
• Controls access to resources while providing the flexibility to choose between and easily modify services and/or configurable options.

Steps
1. Ensure IT organization clearly understands new governance structure and consistently follows the new processes without creating any shadow practices.
2. Check service catalog for configurable options.
3. Check service rules.
4. Define whether the change is standard or non-standard.
5. Plan the change, and schedule it appropriately.
6. Execute the change.
7. Update configuration records.
8. Execute the defined auditing activities per the schedule and in response to any ad hoc requests.

Failure Condition 1
Change fails.

Failure Handling 1
Run back-out script.

Failure Condition 2
Capacity constraint; no landing zone for change.

Failure Handling 2
Delay change, notify consumer of constraint, and find an alternative resource.
Usage Scenario 17 - Autoscale within a Cloud

Actors
Cloud subscriber, cloud provider

Goal
Automatically scale the number of system resources for a subscriber based on current or planned demand. Manage processing of cloud scale requests and provision and track the entire cloud service lifecycle; deliver automated fulfillment for various business requirements and use cases.

Assumption 1
The cloud provider has implemented a scalable architecture capable of automatically “growing or shrinking” based on demand and potentially other parameters, such as end-user experience (for example, transaction response time), regardless of how fast or how slowly the demand changes over time.

Assumption 2
The cloud provider publishes a clear set of business rules for autoscaling with parameters that are set by the subscriber. This includes maximum and minimum resource sizing, and the defined events that trigger growth or shrinkage.

Success Scenario 1
The workload scales out or back based on current environmental conditions.

Steps
1. Front-end site traffic: Scale based on the number of incoming requests, such as web pages, objects, and data transfer, as well as end-user experience, such as transaction response time potentially balanced with business requirements such as budget limitations.
2. The cloud consumer completes the necessary steps to request specific service and request a confirmation for the service provision or expansion.
3. The cloud provider verifies possible constraints of provision request, such as terms of service and contract, and SLAs.
4. Back-end batch processing (scale horizontally): Load-based scaling–scale based on the number of jobs in the queue; time-based scaling–scale based on how long jobs have been in the queue.
5. The cloud provider updates the contract for a specific service and provisions service-specific cloud infrastructure within the terms of service requirements.
6. The cloud provider returns a confirmation message indicating the successful provision of the additional service capacity.

Failure Condition 1
Scale-out request cannot be completed by the cloud subscriber.

Failure Handling 1
Consumer notified of failed request and alternatives are suggested.

Failure Condition 2
Scale-back request cannot be completed by the cloud subscriber.

Failure Handling 2
Consumer notified of failed request and alternatives are suggested.

Usage Scenario 17a - Bursting between Clouds

Actors
Cloud subscriber, cloud provider

Goal
Automatically scale the number of system resources for a subscriber based on current demand, extending to resources from another cloud environment. Manage processing of cloud scale requests and provision and track the entire cloud service lifecycle; deliver automated fulfillment for various business requirements and use cases by extending the system capacity to include additional resources from another cloud environment, on short notice.

Assumption 1
The cloud provider has implemented a scalable architecture capable of automatically growing or shrinking based on demand, regardless of how quickly or slowly the demand changes over time.
Assumption 2
The cloud provider publishes a clear set of business rules for autoscaling with parameters that are set by the subscriber. This includes maximum and minimum resource sizing and the defined events that trigger growth and shrinkage.

Assumption 3
The cloud subscriber has a valid contract in place that supports bursting to an alternate cloud provider, to enable reservation and release of resources on short notice.

Assumption 4
The alternate cloud providers (source and target) are prepared and able to collaborate, have an adequate network between them to enable this, with the necessary security in place, and the SLA regarding service responsibilities are clearly mapped and established.

Assumption 5
An adequate network connection preexist between the original system and the alternate cloud environment, so that compute and storage elements can communicate synchronously, without latency impacts.

Assumption 6
There are two scenarios in which bursting to another cloud is envisioned:

• When the “current” provider is out of resources and cannot accommodate requests to scale-out the service locally
• When the subscriber specifies a ratio or a scaling algorithm that specifies how growth in scaling needs to be balanced between the two providers (for example, the subscriber may want to maintain some kind of a balance for expediting failover processing)

Success Scenario 1
The workload scales out and back based on current environmental conditions, without transaction loss or increased latency, without re-log on action required, and with consistent data afterward.

Steps
1. Bursting triggers:
   - Front-end site traffic: Scale based on the number of incoming requests, such as web pages, objects, and data transfer, as well as end-user experience, such as transaction response time potentially balanced with business requirements such as budget limitations.
   - Back-end batch processing (scale horizontally): Load-based scaling—scale based on the number of jobs in the queue; time-based scaling—scale based on how long jobs have been in the queue
2. The cloud consumer completes the necessary steps to request specific additional resource capacity and accepts a confirmation for the provision or expansion authorization.
3. The cloud provider verifies any possible constraints to provision the request, such as network connectivity, terms of service and contract, and SLAs.
4. The cloud provider updates the records for a specific service, and provisions service-specific cloud infrastructure within the terms of service requirements.
5. The cloud provider returns a confirmation message indicating the successful provisioning of the service.
6. The system begins to allocate transactions to the additional resources at the new provider for processing, with original data remaining in place, and with consistency being maintained per transaction, depending on where it is routed.
7. At the end of the “burst period” the transactions assigned to the “additional resource capacity” are quiesced, and all further transactions are routed to the original infrastructure.
8. The cloud provider returns evidence that any data in memory has been deleted in accordance to the terms of service definitions.

Failure Condition 1
Scale-out requests cannot be completed by the cloud subscriber.

Failure Handling 1
Consumer notified of failed request and alternatives are suggested.

Failure Condition 2
Scale-back request cannot be completed by the cloud subscriber.

Failure Handling 2
Consumer notified of failed request and alternatives are suggested.
Failure Condition 3
Network connectivity is insufficient for additional compute to access remote data and transaction latency occurs.

Failure Handling 3
Consumer notified of problem, and network connectivity updates are suggested.

Failure Condition 4
Transaction data is not consistent after bursting to remote capacity.

Failure Handling 4
Consumer notified to review and update transaction management process, as well as the master data record management process.

Usage Scenario 18 - Comply to Regulatory Requirements

Actors
Cloud subscriber, cloud provider, subscriber regulator, provider regulator

Goal
The regulatory requirements for the cloud subscriber and the cloud provider must be matched and reflected in the SLA between the cloud subscriber and the cloud provider. Contradictory requirements must be clarified, and an agreement must be made that is acceptable to the regulators.

Assumption 1
The regulatory requirements that impact the service are documented and reflected in the SLA for the service. These are corporate requirements as well as jurisdictional requirements. All regulatory requirements to be applied are identified and are part of the contract between the provider and subscriber. Methods for complying with these requirements have been identified and agreed upon between provider and subscriber. Metrics to provide evidence have been agreed upon by the regulators.

Assumption 2
Both the cloud subscriber as well as the cloud provider are willing to comply with all regulatory requirements, irrespective of whether they come from the subscriber regulator or the provider regulator. If the regulator’s requirements of both the cloud provider and cloud subscriber do not dictate in which premises the data must be retained (either the provider’s or the subscriber’s) for a defined time, (such as seven to 10 years for the financial industry) it is up to both to agree on this as part of their contract.

Success Scenario 1
Termination of the service is compliant with regulatory requirements. All components of the service are shut down. All data in files or databases is frozen and run down according to the defined contractual agreement. All regulatory reports are generated to provide evidence of the compliant termination of the service and required data storage or deletion.

Steps
1. The service is terminated, and the termination does not leave any open transaction or unresolved operation.
2. Open files or databases of the service are closed or shut down. The audit trails and log files are closed.
3. The audit trail is written to its long-term storage location (or archive); the service logs are written to their near or mid-term storage location, such as a monitoring system, disk, or archive, according to the SLA for the service.
4. The operational data\(^\text{12}\) of the service is transferred to the place defined in the SLA, such as downloaded to the cloud subscriber or in some way archived, for retention (see Assumption 2).
5. For assurance levels Silver, Gold, and Platinum, all files and databases of the operational storage, such as disks and solid-state drives, are deleted securely, such that the data is not recoverable.\(^\text{13}\) Additionally, for assurance levels Gold and Platinum, the file systems allocated to the service are reformatted, including overwriting all physical disk blocks.
6. For assurance levels Gold and Platinum, the memory of all machines (physical and virtual) and in the various network components, such as routers, switches, and firewalls, is securely deleted. The technical deletion process must be defined as being technically feasible and approved.
7. All archives on disks or tapes of the service are deleted securely so that they are either not recoverable\(^\text{14}\) or destroyed.
8. The provider hands over a report evidencing all cleanup actions that were executed. The report content is defined and is part of the contract between subscriber and provider. The report takes the assurance levels into account.

---

\(^\text{12}\) All data directly linked to a service is considered operational; that is, business data, logs, and audit trails. Configuration data and user-related data (identity, entitlements) are not considered to be operational data.

\(^\text{13}\) Crypto shredding might be considered a possible option for non-recoverable deletion.

\(^\text{14}\) “Not recoverable” means possible only with far too high an effort or not recoverable at all (this might depend on the specific case).
Failure Condition 1
Some transactions cannot be closed; some operations cannot be quiesced or resolved when the service is shutting down gracefully.

Failure Handling 1
The service is forced to shut down; unresolved operations or open transactions are logged and provided to the subscriber.

Failure Condition 2
The subscriber’s database or some files are corrupted after the service’s shutdown.

Failure Handling 2
Application log and database transaction log are provided to the subscriber with the database files. Application log and corrupted files are provided to the subscriber as well.

Failure Condition 3
The transfer of the operational data to the subscriber failed.

Failure Handling 3
The transfer is repeated, possibly over a different channel.

Failure Condition 4
The provider cannot provide any evidence report that fits the contractual requirements between subscriber and provider.

Failure Handling 4
The provider has to redo some cleanup actions and reproduce an evidence report. For assurance levels Gold and Platinum, the provider has to allow an external auditor to verify the completion of the cleanup process.

Usage Scenario 19 - Service and Data Termination and Deletion

Actors
Cloud subscriber, cloud provider, subscriber regulator, provider regulator

Goal
Complete the use of a service, terminate it, write out the relevant data according to requirements, scrub the infrastructure, and return the resources to available pools.

Assumption 1
A termination framework and process has been agreed upon within the contract for services or service elements, and a formal request for a service termination exists based on this.

Assumption 2
Data classification exists, with defined processes and rules.

Success Scenario 1
The service is ended as expected, data is handled as expected, and the resources are returned to the available pools for reuse.
Steps
1. The service is terminated, and the termination does not leave any open transaction or unresolved operations.
2. Open files or databases of the service are closed or shutdown. The audit trails and log files are closed.
3. The audit trail is written to its long-term storage place, such as an archive; the service logs are written to their near or mid-term storage location, such as a monitoring system, disk, or archive, according to SLA for the service.
4. The operational data\textsuperscript{15} of the service is transferred to the place defined in the SLA and downloaded to the cloud subscriber, (or archived) for retention. See Assumption 2.
5. For assurance levels Silver, Gold, and Platinum, all files and databases from the operational storage unit are deleted securely and are not recoverable.\textsuperscript{16} Additionally, for assurance levels Gold and Platinum, the file systems allocated to the service are reformatted, including overwriting all physical disk blocks.
6. For assurance levels Gold and Platinum, the memory of all machines (physical and virtual) and in the various network components, such as routers, switches, and firewalls, is securely deleted as well. The technical deletion process must be defined as being technically feasible and approved.
7. All archives on disks or tapes of the service are deleted securely and are not recoverable or are destroyed.\textsuperscript{17}
8. The provider hands over a report evidencing all cleanup actions that were executed. The reports content is defined and is part of the contract between subscriber and provider. The report takes the assurance levels into account.
9. The resources are returned to the available pools for re-allocation within the cloud.

Failure Condition 1
Some transactions cannot be closed, or some operation cannot be resolved when the service is shutting down gracefully.

Failure Handling 1
The service is forced to shut down, unresolved operations or open transactions are logged and provided to the subscriber.

Failure Condition 2
The subscriber's database or some files are corrupted after the service's shutdown.

Failure Handling 2
Application log and database transaction log are provided to the subscriber with the database files. Application log and corrupted files are provided to the subscriber as well.

Failure Condition 3
The transfer of the operational data to the subscriber failed.

Failure Handling 3
The transfer is repeated, possibly over a different channel.

Failure Condition 4
The provider cannot provide an evidence report that fits the contract between subscriber and provider.

Failure Handling 4
The provider has to redo some cleanup actions and reproduce an evidence report. For assurance levels Gold and Platinum, the provider has to allow an external auditor to verify the completion of the cleanup process.

\textsuperscript{15} All data directly linked to a service is considered operational; that is, business data, logs, and audit trails. Configuration data and user-related data (identity, entitlements) are not considered to be operational data.

\textsuperscript{16} Crypto shredding might be considered a possible option for non-recoverable deletion.

\textsuperscript{17} “Not recoverable” means possible only with far too high an effort or not recoverable at all (this might depend on the specific case).
SERVICE ORCHESTRATION ARCHITECTURAL OVERVIEW

Service orchestration focuses on the triggering of a number of workflows through different layers of the cloud ecosystem as described earlier in the Introduction. It is therefore helpful to conceptualize the minimum participant technology and systems comprising service orchestration in context of the workflow activity within a cloud ecosystem.

Cloud services are based on a number of enabling technologies that allow service to be delivered according to the characteristics defined for the cloud as defined by the National Institute of Standards and Technology (NIST). Focus in this master usage model (MUM) is on the interactions to the cloud subscriber. But it must also be noted that there are similar interactions with the service provider as well as with other solution providers within the overall service as recognized in the Services Interface Overview of this document. Figure 2 depicts a high-level, generic cloud ecosystem architecture that illustrates some of the service orchestration key elements while also considering both hard and soft interfaces.

A provision exists for an organization or organizations to take on a number of roles as middlemen, usually described as brokerage, integration, aggregation, or orchestration. Large parts of these functions might be automated, leveraging APIs and cloud brokers, with agents. Open API-based interactions leveraging representational state transfer (REST) protocols are strongly recommended for all of these interactions.

The orchestration process itself must belong to the cloud consumer, who has overall responsibility for triggering requests and paying for services. However, the orchestration systems and tooling can be provided by the consumer, or one of the cloud provider parties. The orchestration process from the cloud provider side can also initiate a process or processes in the cloud consumer side: for example, notification or alerting. These parties could include their own IT organization or a third-party cloud provider who is also prepared to act as an intermediate to other cloud providers. When multiple third-party cloud providers are involved, more and more ownership and responsibility defaults to the cloud consumer, especially when the consumer requires the ability to move or burst services between providers, based on various business criteria and usage scenarios. These responsibilities include data management, transaction management, security, and integration between the providers and themselves, and federation of service operational levels of agreements, to achieve SLAs.

Figure 2. Generic cloud ecosystem architecture.
In the context of bursting, long-distance migration, and many of the other cloud usage scenarios, the consumer at a minimum needs to take care of the following:

- Initiation of change request via the service orchestration system
- Security domain and federation including encryption keys
- Configuration management of the system allocated to the cloud location including data and master data record management
- License management and entitlement
- SLA and service requirements definitions

The systems with which the orchestration process must interact to enable these responsibilities are highlighted in the blue area of Figure 2.

The interactions to and from cloud consumers and to and from cloud providers are similar, with central functionality sometimes acting primarily as a switching center. They may be invoked through a portal, but are able to be used in a more sophisticated manner when invoked by APIs. One issue that must be considered in such a multi-party arrangement is how visible, in terms of service levels, prices, and volumes, the different suppliers’ services will be between each other and to non-customers, and what level of abstraction (via the public view of the service catalog) will satisfactorily mitigate this potential concern.

The components addressed and enabling the interactions are as follows:

- **Master services agreement.** As defined by the ODCA Usage Model: Regulatory Framework, a contractual framework is identified according to which services can be commissioned based on predefined requirements, terms, and conditions.

- **Services catalog.** As defined by the ODCA Usage Model: Service Catalog, a standard framework for the description and publishing of cloud services such that they can be described and discovered in a standard way (and potentially compared between suppliers).

- **Configuration.** Complements the services catalog by providing an indication of the available capacities and prices of the supplier’s environment at any one time. This can be used to support the implementation of conditional pricing, volume discounting, or spot market, a market where services are traded for immediately. Additionally it enables the customization of standardized services, based on predefined service variables (for example, the number of users)

- **Provisioning.** Determines the best routing and fit for any particular request and assigns it to the relevant suppliers or subsystem. This can be established by predefined criteria as described in TREC from the EC-sponsored OPTIMIS project.

- **Image library.** Used to hold and allocate reconfigured machine images for deployment across any supplier. Images can be deposited by cloud consumers or providers, and may or may not be made available to others.

- **Security systems including identity management.** Deployed on a federated basis, using standard protocols, such as SAML, such that cloud consumer organizations maintain their own directory of their users, and cloud providers can establish trust relationships to permit access to their environments. These functions can include both users who are entitled to commission and configure systems as well as the end-users themselves. It includes aspects such as authorization levels to order further facilities.

- **Service operations.** Takes place mostly with the cloud provider who delivers services to the relevant cloud consumers. In addition, some other possible services are shown that surround various operational services:
  - **Continual service improvement.** Helping a (potential) cloud consumer determine what systems are most suitable for cloud, how they should be configured, and on which platforms they can be deployed.
  - **Service transition.** Actually converting applications for cloud platforms, identifying and transferring the necessary data, setting up the environment, and so on; these are tasks that the cloud consumer would otherwise have to do for themselves.
  - **APIs.** A set of routines, protocols, and tools for building software application interactions. A good API makes it easier to address a system or develop a program by providing all the building blocks.

Successful IaaS service orchestration addresses multiple systems and processes, as illustrated in the architectural concept, including a service catalog, a services interface, and KPIs. The following sections cover each of these elements in turn. The Service Catalog Overview includes background information and definitions as well as a detailed lifecycle process for users. Information about a service catalog’s structure also is included.

---

18 See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)
19 See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)
20 TREC from the EC-sponsored OPTIMIS project. [www.optimis-project.eu/content/ec-optimis-project-release-first-open-source-cloud-toolkit-service-providers](http://www.optimis-project.eu/content/ec-optimis-project-release-first-open-source-cloud-toolkit-service-providers)
Machine-to-Machine Orchestration in the Context of API Communication and Orchestration Architecture

Context
The principle guidelines behind machine-to-machine (M2M) communication in service orchestration are in the context of interoperability, maintainability, and extensibility, at the machine level, by and between machines. To enable this, APIs are extensively used. APIs in the context of cloud, and particularly service orchestration, and the appropriate standards, are explained in some detail later in this document.

Actors
Cloud providers, cloud subscribers, cloud brokers

Every actor’s activity is dynamic:

- Cloud subscriber may consume new services, change current consumption model, move or add services to/from another cloud provider, and so on, and hence change the orchestration process.
- Cloud providers can modify the orchestration processes and rules, the service catalog semantic description, and the way back-office services work (see Figure 3).
- Cloud broker/s can create an orchestration process between several cloud providers and subscribers.

Figure 3 illustrates the M2M communication concept and requirements.

![Figure 3](image_url)

Orchestration Requirement
The orchestration application/s and process designers require a machine-readable description that includes the following aspects (functional and non-functional requirements):

- The service(s) from the service catalog
- Cloud provider functionalities and orchestration description as described in the ODCA Usage Model: Service Orchestration and shown in Figure 2. Generic cloud ecosystem architecture.
- The protocol (typically encapsulated within an API) to communicate with the cloud provider’s self-service portal

A test (and maybe development) environment is needed in order to check the developed orchestration application/s and process and to test the deployment of each service element being added via the orchestration application/s and processes. This is particularly relevant where a consumer may want to orchestrate their own application deployment onto infrastructure provided by a cloud provider, possibly in addition to elements already available from that provider (for example, a database service). Full testing of such a “hybrid” system capability for automatic deployment is essential, up to and including the network elements.

---

21 See [www.opendatacenteralliance.org/library](http://www.opendatacenteralliance.org/library)
Requirements
In order to have an interoperable, maintainable, and extensible M2M process, the following is needed:
• Clear separation between lifecycle phases/layers.
• Those layers should be managed with all aspects of application lifecycle management such as versioning, backward compatibility, and change management.

The following should be common to all cloud providers and subscribers:
• Design, development, and QA phase. At the design phase, a common machine-readable semantic and syntax is needed to describe the above-mentioned system element requirements, which will enable successful orchestration.
  – XML and XSD can be used as a mechanism to define and validate the syntax.
  – A publishing process for those definitions is needed from the cloud provider.
  – A translation of the common semantics to a cloud-specific syntax to be used by the proper APIs (see below).

This layer may be cloud-provider specific:
• Runtime service orchestration phase. At runtime, a set of APIs is needed to create a valid orchestration process between one or more subscribers and one or more providers/brokers. This process needs to bridge the required security and operational domains between the actor’s technical systems according to predefined frameworks.
  – Maintain a “workflow process engine” between all parties to handle the orchestration process and the various work packages that are part of that workflow.
  – Maintain individual “tasks/work packages” that are part of the workflow that create a service, and monitor their status in the various subsystems; for example, create VLAN/LUN/Authorization/Image Encryption as sub-tasks of a service work package named “VM allocation.”
  – Manage application-level tasks and work packages such as authentication, authorization, service-level error code returns/messages, timeouts, and human intervention when needed.

This layer is common for all actors, although there might be some APIs that are relevant for only one actor (for example, an “update subscriber account” may be valid for a cloud provider only).

The following layer should be common to all cloud providers and subscribers:
• Runtime operations phase. At runtime, a protocol between one or more subscribers and one or more providers is needed. This is a “service level” protocol rather than a “communication level” (for example, TCP/IP, HTTP(S), VPN).

The protocol handles issues such as the following:
  – A reliable connection (that is, one with atomicity, consistency, isolation, and durability (ACID) properties) at the service level
  – Service naming convention, advertising, and discovery
  – Messaging management: bundle together several requests from the API layer into one message, keep the right sequence between messages, treating messages in sequential or parallel order, and so on
  – Security metadata including rights, role privileges, and access management

This layer is common for all actors, although there may be some APIs that are relevant for only one actor (for example, “discover a service” may be valid for only the cloud broker and cloud subscriber).
Service Catalog Overview

Service catalogs are designed to help organizations in their assessments and selections of cloud services. A detailed and up-to-date service catalog is an essential component of any provider's offerings. It is the mechanism by which a consumer can determine the capabilities and characteristics of one or more services in an industry-accepted standard and consistent manner. The ODCA Usage Model: Service Catalog is aimed at shaping the best practices and mechanisms that should be included in service catalogs such that potential subscribers are better equipped to perform a more precise and programmatic discovery of comparable services.

The access to a service catalog must be maintained in a secure manner. Access to a private or public cloud should always have the appropriate levels of security. Public clouds may have lessened security controls due to the fact that a publicly accessed catalog may be browsed by all. On the other hand, a private (or hybrid) cloud solution requires that access be through secure, encrypted channels. Any cloud catalog solution that requires that the requestor be authenticated will need to perform this authentication through secured channels such that credentials are not compromised. As mentioned previously, a public cloud catalog typically allows anyone to browse the services without authentication, and often through non-secured channels. On the other hand, a private or corporate cloud catalog will require user authentication prior to catalog access, and the cloud catalog is secured from external, unauthorized access.

In the case of a private (and many times, public) cloud catalog, roles are defined for the different user groups such as general user, approver, financial approval, IT administrator, and service managers. A general user may have access to only browse a catalog service, whereas an IT administrator or service manager may view all service requests, status, owner, and time incurred since request. This role-based access provides only the appropriate information to the appropriate roles and those authorized to view that information. The authentication and access rights to relevant sections of a service catalog should be on only a “need-to-know” basis, and areas such as the collection of personal information and payment data will always require highly secured communication channels.

In many cases, once a cloud service is requested and approved, if required, the provisioning of the service may require access to subsystems or provider solutions in order to provision the newly requested service. Communication from the cloud service catalog to any external subsystems or external provider solutions requires encrypted channels to secure the communication. It is advised that even a cloud service catalog that is behind a corporate firewall, and all subsystems within the same firewall, is also encrypted to further secure the system from insider threat and potential system penetration.

After new services have been provisioned, access to newly provisioned services and to service reporting should require the appropriate access controls including user identification and authentication. Access to the service catalog of a cloud subscriber by a different (unauthorized) cloud subscriber should be explicitly prevented.

Depending on the consumer’s business service IT offering to their consumer community, multiple means of interacting with the provider services are needed. In the simplest model the subscriber uses the provider service catalog offering as-is and does not aggregate services from multiple providers. The provider solution produces reports, provisions new service requests or changes to existing services, and performs all required process automation activities in support of provisioning, maintenance, change requests, and interaction with the multiple subsystems (that is, billing, configuration management databases (CMDBs), credit card systems, security systems, and so on). Once the subscriber catalog has been instantiated, the consumer has access to the catalog solution to easily request and utilize provider services with minimal effort.

In the case of a comprehensive service solution whereby the catalog is a bundling of discreet or interdependent provider services (often from multiple providers), a level of meta-data standardization is needed such that a subscriber may easily retrieve all the necessary data related to a service. This service metadata will need to be incorporated into the consumer’s catalog taking into account the following:

- Commercial aspects are addressed such that contractual terms are adhered to and cost detail is provided.
- Technical detail of the service, such as capacity limits, performance expectations, and bandwidth, is made available.
- Service offering specifications include:
  - All information regarding the service, including service description, optional add-on services, and service level expectations, are published.
  - Questions required to fulfill a service request are presented along with valid answer options.
  - Dependencies upon another service (or services) are clearly defined and identified.

The concept and design for the service catalog should be applicable to all layers of cloud services, from IaaS to PaaS and SaaS, and to other emerging services such as a virtual private data center as a service (VPDaaS), data as a service (DaaS), and other emerging services (“X”aaS). The desire is that there should be a common data model (in a self-describing format such as XML) that will help drive widespread adoption by

---

22 See www.opendatacenteralliance.org/library
providers and subscribers to access the service catalog using one standard data model interface. The structure may be different for the various service types—IaaS, PaaS, SaaS, and so on—and will need to be flexible enough to accommodate new capability and service features as they materialize.

In order to more clearly illustrate the orchestration process, the following usage scenario is offered for reference purposes and document alignment. Specific notes should be taken of the service elements, service catalog aspects, and people-based services aspects. This example is a small sample of the orchestration process and not meant to be all inclusive.

Example
In this scenario, the subscriber has assembled a service catalog for a number of PaaS offerings. The service offerings are essentially a set of bundled provider services requiring detailed coordination across multiple provider services. The subscriber has predetermined which provider solutions will compose each PaaS offering (and for which each of them is atomically responsible), as well as how options will be presented to the consumer (for which the service subscriber takes overall responsibility and coordination leadership among the contributing providers), service pricing, and optional add-ons. The subscriber is responsible for orchestrating the service provisioning and interacting with each provider to fully provision a bundled service request. In addition, the subscriber must identify and set up the service transfer points, thereby delineating the points where responsibility for service components transfers between all participating parties, and where atomic-level responsibility starts and ends. Each service provider in this example is responsible for its own monitoring and reporting, and the subscriber is responsible for consolidating these reports and coordinating between the contributing providers in the event of an incident or problem (for example, a security breach). The provisioning of one PaaS request may require service orchestration across three or four provider solutions. A provider solution may have specific dependencies in order to fulfill a request such as access to an available infrastructure or IaaS in order to provision the platform solution.

To illustrate, let’s look at a sample service orchestration required to provision a PaaS, in this case a database platform.

A consumer searches the service catalog for a service that matches specific database requirements. The consumer views each service offering along with a detailed description, options on compute, network, storage resources, available locations, pricing, estimated time for provisioning, and service level options.

• The consumer selects the database platform that best meets the requirements and begins providing the required information for the service request. The database platform that has been requested will be delivered by Provider A.

• Provider A provides the database platform, but requires an existing infrastructure. The consumer must identify an existing infrastructure or select an option to provision new infrastructure (which is added to the service cost). This consumer does not have infrastructure, so selects the option to provision infrastructure. This will be delivered by Provider B.

• This service offering provides three service tiers: Silver, Gold, or Platinum. The Gold level is selected which includes commitments such as up to 10 min unavailability; real-time service reporting, and continuous monitoring. In this scenario the continuous monitoring will be provided by Provider C and this cost is added to the overall service cost.

Once the consumer completes the order, the service orchestration process begins. Looking behind the scenes, this specific PaaS is composed of interdependent services from multiple providers. Provider B provides infrastructure (IaaS), Provider A provides the requested database platform, and Provider C provides continuous system monitoring and remediation. There may also be multiple providers that offer IaaS, multiple providers with various PaaS, and possible multiple system monitoring providers. Based on the requirements of this service request, the bundled services from Provider A, Provider B, and Provider C are needed to fulfill the PaaS service request. Furthermore the platform must be provisioned first by Provider B, and the access to this platform must be provided to Provider A such that the platform may be provisioned to the newly provisioned infrastructure, and finally authorized access to the database application, the cloud VMs, and the OS will be provided to Provider C to begin the monitoring.

This is an example of service orchestration that involves careful coordination of activities across multiple provider solutions and systems. In addition to the aforementioned details, the subscriber must include processes to update the CMDB, correlate invoicing composed of vendor service pricing plus subscriber costs, and roll up metrics, SLAs, and reports into a comprehensive consumer dashboard.
Logical Architecture

• The logical architecture for cloud service orchestration provides a more detailed structure defining the activities and services that must be performed for the subscriber requests. The logical architecture includes the business services, the business systems, and the communications between the layers or tiers of the model.

• Figure 4 depicts service orchestration and the required interactions and activities of the three logical layers: user interface, business services, and technical services.

Initiation

The consumer accesses the service catalog and browses cloud service offerings from multiple providers. The consumer browses items such as the service description, pricing, prerequisites and dependencies, payment options, service level options, and contractual terms.

After considering the various services, the consumer selects the cloud service offering of interest and begins the request process by providing the required detail needed to submit the request.

A completed service request is transferred to the service orchestration to begin the request fulfillment process. Service orchestration uses the appropriate APIs to communicate with the appropriate provider service. Activities are coordinated and executed, which include request validation, reserve cloud service, provision new service, notify consumer of status (email or other means), update service catalog dashboard with status of service request, enable secured access for designated users, and commence operations and support.

Although Figure 4 shows service orchestration occurring in the business tier, the reality is that much of the service orchestration will likely be performed by the service provider at the technical tier. A high-level request may be made by the business tier to the technical tier, such as a new cloud request that transfers control to the technical tier using an API call to provision a new cloud storage request. The coordination of the service request and activities such as initiating IT operations and support, change management, and updates to the CMDB, may be orchestrated within the technical tier. A realistic initiation process has an orchestrated business layer that relies on orchestrated processes within the technical tier.

Figure 4. Logical layers of service orchestration.
Run, Monitor, Change
The following is an example of the cloud service run, monitor, change phase and the required interactions and activities of the three logical layers: user interface, business tier, and the technical tier.

A consumer has previously requested and been provided access to one or more provider services. Additionally the consumer received notification and instructions on accessing, reporting, and requesting changes to said service.

The consumer is able to access the service catalog in order to utilize the provisioned service, monitor activity, and access reports and metrics. The consumer is provided service monitoring capability such as health, performance, and availability monitoring, capacity planning, usage data, and billing detail. In addition, the consumer may request audit information and may request changes to the services, items such as starting or stopping the service, or increasing capacity.

Any changes to the service are initiated as a service request at the user interface tier and traverse through the business and technical tiers, similar in nature to a new service request.

The service provider performs overall operations support (such as billing, incident management, and service desk) and IT operations (such as system administration and event management) support at the technical tier.

Termination
The following is an example of the cloud service termination phase and the required interactions and activities of the three logical layers: user interface, business tier, and the technical tier.

The consumer has made a decision to terminate a previously provisioned service, the contractual agreement is terminated by one of the parties, or a service subscription has hit the defined expiration date. This termination may initiate in the user interface layer as a request or alternatively may be triggered within the business tier as part of the service orchestration process. As part of the termination, the provider triggers a “default” data archival process as defined (if defined) to comply with the appropriate compliance requirements for that data class, which interfaces with the technical tier to invoke a contract exit and termination clause, end billing and produce a final bill, release resources and return to the available resource pool, and notify the IT support and operations teams and processes of service termination.

Services Interface Overview
In addition to a services catalog, another element in a successful service orchestration is the services interface, which includes APIs, the command line interface (CLI), or graphical users interfaces (GUIs) that are used when orchestrating a cloud service. The need for such interfaces was demonstrated in the Service Orchestration Architectural Overview, and particularly in context of the need to enable M2M interactions. For this purpose, standardized instructions are needed, which are translatable between environments, to perform standardized tasks. In some cases, a standardized framework may be used and tasks defined within that framework, but these controls then have to be published to authorized parties, to enable their interactions. The services interface provides the means for the cloud subscriber to create IaaS compositions in order to automate and integrate the capabilities within business workflows. Other tools, such as CLIs or GUIs, are also important, but APIs are ideal for cloud subscriber orchestration, especially in the M2M level layer, but also for all sub-element interactions according to a standardized framework.

In order to dynamically adopt and operate cloud-based services, these service interfaces must be capable of integration into the service consumers’ automation systems, preferably without human involvement, considering the workflow and approval stages. This means that the service interfaces need to be open so as to not limit potential cloud users by means of special license or cost limitations on the use of the interface, and to enable the automated working of systems and services.

The interfaces need to interact with standardized work flows and service orchestration triggers in a consistent and predictable fashion, globally, and according to defined service qualities. This references the impact on each of the configuration items (CI) in the service catalog, the ordering and operations processes, and the security and compliance of each CI. Should any interfaces be restricted or have special licensing or other prerequisites, the openness and adoptability of the overall cloud service comes under risk. Any limitations would potentially restrict the potential user base of the cloud service and represent lock-in or deviation from open standards. In addition, maintenance and ongoing development of the service interfaces must be non-intrusive, enabling sustained operations without impact or a requirement for downtime.

The web services invoked using the services interface should be written in a manner consistent with cloud computing practices, comprehending basic characteristics such as on-demand/self-service, elasticity, and multi-tenancy. For example, subscribers should be able to call APIs in massive scale. Instead of failing requests when capacity is low, a rate-limiting technique throttles requests, returning an error only when
resources are completely consumed. Other principles of cloud-aware application design that should be applied to web services are outlined in the best practices paper *ODCA Developing Cloud-Capable Applications*.23

This section builds upon several ODCA Usage Models: *Compute IaaS*,24 *Standard Units of Measure for IAAS*,25 and *Guide to Interoperability Across Clouds*.26 It takes the perspective of the cloud subscriber in that the APIs are derived from the usage scenarios defined earlier in this document. APIs are expressed in terms of the representational state transfer (REST) architectural model. The intent is not to provide a complete specification for cloud providers. Rather, it is intended to convey the set of capabilities required in an API.

With REST, there is the concept of resources and collections. A resource is a typed object with properties, methods, and associations with other objects. A collection is a directory or group of resources. Since resources are addressed through Uniform Resource Identifiers (URIs), the expectation is for the cloud provider to use conventional URI design to publish APIs. For example, for a provider named “myprovider” and a subscriber named “mysubscriber”:

- http://api.iaas.myprovider.com
- http://api.iaas.myprovider.com/services/V1
- http://api.iaas.myprovider.com/mysubscriber
- http://api.iaas.myprovider.com/mysubscriber/servicerequests

In addition, techniques for searching and filtering collections should be supported. For instance, the selection of a certain subset or number of items within a collection can be accomplished by applying a “$filter” predicate expression as part of the URI.

Another important REST concept is the use of standard methods. The basic four methods are as follows:

- Post - Create a resource
- Get - Read information about a resource
- Put - Update a resource
- Delete - Remove a resource

Table 2 describes the requested service orchestration collections and associated usage scenarios. Table 3 describes the requested resources and maps them to the associated usage scenarios.

Table 2. Requested collections and associated usage models.

<table>
<thead>
<tr>
<th>Collections</th>
<th>Post</th>
<th>Get</th>
<th>Put</th>
<th>Delete</th>
<th>Usage Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>NA</td>
<td>Get a list of all available services</td>
<td>NA</td>
<td>NA</td>
<td>1. Compose service</td>
</tr>
<tr>
<td>Roles</td>
<td>NA</td>
<td>Get a list of all possible user roles</td>
<td>NA</td>
<td>NA</td>
<td>1. Compose service</td>
</tr>
<tr>
<td>Servers</td>
<td>NA</td>
<td>Return a list of all compute objects that have been created through the provisioning process</td>
<td>NA</td>
<td>NA</td>
<td>7. Start dependencies 8. Stop dependencies 9. Suspend cloud service 10. Resume cloud service</td>
</tr>
<tr>
<td>Events</td>
<td>NA</td>
<td>Get a list of possible events that can be generated by the system</td>
<td>NA</td>
<td>NA</td>
<td>12. System administration and remediation</td>
</tr>
<tr>
<td>Key Performance Indicators (KPIs)</td>
<td>NA</td>
<td>List the KPIs that are supported in the IaaS service</td>
<td>NA</td>
<td>NA</td>
<td>13. Reporting</td>
</tr>
<tr>
<td>Reservations</td>
<td>NA</td>
<td>Return a list of all reservations for the account or user</td>
<td>NA</td>
<td>NA</td>
<td>3. Reserve resources for service</td>
</tr>
<tr>
<td>Servicerequests</td>
<td>NA</td>
<td>Return a list of service requests that have been created for the user or account</td>
<td>NA</td>
<td>NA</td>
<td>2. Submit provisioning request</td>
</tr>
</tbody>
</table>

---

23 See www.opendatacenteralliance.org/library
24 See www.opendatacenteralliance.org/library
25 See www.opendatacenteralliance.org/library
26 See www.opendatacenteralliance.org/library
### Table 3. Requested resources and associated usage scenarios.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Post</th>
<th>Get</th>
<th>Put</th>
<th>Delete</th>
<th>Usage Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Subscriber</strong></td>
<td>NA – Must contact the provider to create the account</td>
<td>Get current subscriber settings</td>
<td>Update subscriber settings</td>
<td>NA - Must contact the provider to delete the account</td>
<td>1. Compose service 16. Change a deployed service instance</td>
</tr>
<tr>
<td><strong>User</strong></td>
<td>Provision a new user for the subscriber account of a particular role</td>
<td>Get current user settings</td>
<td>Update user settings</td>
<td>Delete a user</td>
<td>1. Compose service 12. System administration and remediation</td>
</tr>
<tr>
<td><strong>Reservation</strong></td>
<td>Reserve resources for a service request with expiration date</td>
<td>View current resource reservation for a service request</td>
<td>Update resource reservation; for example, the expiration date</td>
<td>Remove reservation for a service request</td>
<td>3. Reserve resources for service 6. Reopen expired request</td>
</tr>
<tr>
<td><strong>Server</strong></td>
<td>Boot a compute object after successful reservation and initialization</td>
<td>Return the state associated with a compute object</td>
<td>Change the state of the object to trigger reboot, suspend, resume</td>
<td>Shutdown and release reserved resources</td>
<td>7. Start dependencies 8. Stop dependencies 9. Suspend cloud service 10. Resume cloud service</td>
</tr>
<tr>
<td><strong>Eventlog</strong></td>
<td>Create an event log object</td>
<td>Return trend data of monitors for a particular time period – apply filter</td>
<td>Update event log properties</td>
<td>Remove event log object</td>
<td>11. System monitoring and data collection</td>
</tr>
<tr>
<td><strong>Forecast</strong></td>
<td>Create a forecast object for a particular time period, based on history, KPIs, and forecast properties</td>
<td>Generate reservation settings based on the forecast object</td>
<td>Update forecast properties</td>
<td>Remove forecast object</td>
<td>14. Capacity planning</td>
</tr>
<tr>
<td><strong>KPI</strong></td>
<td>NA</td>
<td>Get the details for a particular KPI—current value—not just the metadata</td>
<td>NA</td>
<td>NA</td>
<td>13. Reporting</td>
</tr>
<tr>
<td><strong>Provisionqueue</strong></td>
<td>Submit a service request to the provisioning queue</td>
<td>See a list of service requests that have been submitted for provisioning</td>
<td>Remove request from the provisioning queue</td>
<td>Delete a service request that is not fully deployed</td>
<td>2. Submit provisioning request 5. Track status and manage deployment</td>
</tr>
<tr>
<td><strong>Remediation</strong></td>
<td>Attributes: 1. The Monitor with which the remediation is associated. 2. The event for which remediation will take place 3. The remediation script</td>
<td>List the current remediation for a monitor</td>
<td>Change any remediation attributes for a monitor</td>
<td>Remove remediation for a monitor</td>
<td>12. System administration and remediation 17. Autoscale within a cloud</td>
</tr>
<tr>
<td><strong>Report</strong></td>
<td>Create a report, specifying the KPIs and time period</td>
<td>Get the data associated with a particular report that has been created, including the location of where the report is published</td>
<td>Change attribute in order to trigger the creation of a report – specify the report date/time or recurring frequency and report delivery mechanism</td>
<td>Remove a report</td>
<td>13. Reporting 15. Auditing</td>
</tr>
<tr>
<td><strong>Reservation</strong></td>
<td>Reserves resources for a service request with expiration date</td>
<td>View current resource reservation for a service request</td>
<td>Revise a reservation</td>
<td>Remove reservation for a service request</td>
<td>3. Reserve resources for service</td>
</tr>
<tr>
<td><strong>Service Request</strong></td>
<td>Set up an order for a new service</td>
<td>View detail of a specific service request</td>
<td>Change the settings of a service request that is in progress</td>
<td>Delete the service request</td>
<td>1. Compose service 5. Track status and manage deployment</td>
</tr>
<tr>
<td><strong>Health</strong></td>
<td>NA</td>
<td>Retrieve general information describing whether the service is up and functioning normally</td>
<td>NA</td>
<td>NA</td>
<td>All</td>
</tr>
</tbody>
</table>
Finally, APIs are designed to function in various “directions,” and these are approximately “mapped” as follows in the context of the element in focus (that is, infrastructure would consider PaaS to be its client, while PaaS would consider IaaS to be its technology direction):

- **North.** Consumer API for consumption of the element in focus’s services
- **South.** Infrastructural or technology API for underlying sub-element communication
- **East and West.** API Interfaces to other parallel systems; for example, for infrastructure it might be to burst into other cloud environments or even to parallel security enablement systems

**KEY PERFORMANCE INDICATORS OVERVIEW**

KPIs are control tools designed to allow the measurement of the following aspects of a service:

- Progress
- Compliance
- Effectiveness
- Efficiency

These measurements are critical in order to control and improve the process and hence the service(s) delivered by the process and their executed steps.

It is important to understand that KPIs are defined and employed within the context of the metrics tree, as described diagrammatically in Figure 5.

![Figure 5. Metrics tree.](image-url)
A defined KPI has the following basic elements:
- A basic and clear statement of the KPI
- A statement to justify the definition and use of the KPI
- A formula for the calculation of the KPI
- Suggested targets and threshold percentages
- A statement if within targets
- A statement if below/outside of targets

For a generic overview of KPIs and their composition and definition, please refer to the ODCA Standard Units of Measure Usage Model.27

As shown in Figure 5 and within the context of this Service Orchestration Usage Model, the vision of service orchestration is suggested as thus:

“To support the effective and efficient discovery (by a cloud service subscriber) and orchestration (by a cloud service provider) of (discovered and offered) services to meet the stated requirements of a cloud service subscriber”

In order to measure, control, and improve the services and processes associated with service discovery and orchestration, both cloud service subscribers and providers need to define and employ an appropriate set of KPIs. Hence, within the context of service orchestration, the following basic KPIs are suggested to control and measure the key aspects of a published, composed, and orchestrated service:

- Service availability
- The speed and timeliness of execution
- The accuracy of the expected outputs
- The compliance of the orchestrated service output(s) with applicable legal and regulatory constraints
- The (financial) cost service orchestration/execution/decommissioning
- Cloud subscriber satisfaction
- Service execution incident business impact ratio

The following tables describe these suggested KPIs for service orchestration.

**Service Availability**

Table 4 describes the suggested KPIs for service availability.

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>Average availability of all services orchestrated by a cloud service subscriber.</td>
</tr>
<tr>
<td>KPI Justification</td>
<td>It is critical to establish the ongoing availability metrics for all services orchestrated by a cloud service subscriber to ensure that impacts from non-service availability can be addressed as soon as possible and mitigated from future occurrence.</td>
</tr>
<tr>
<td>KPI Calculation Formula</td>
<td>(Number of orchestrated services meeting availability targets × 100%) ÷ Total number of orchestrated services</td>
</tr>
</tbody>
</table>
| Suggested Targets     | • Target = 100%  
                          | • Threshold = 90% |
| If Within Targets     | Orchestrated services are available at levels that meet business needs.         |
| If Below Targets      | The services provided by the cloud service provider(s) are providing little value and, in extreme cases, could be putting business revenue (of cloud service subscriber) and reputation at risk. |

27 See www.opendatacenteralliance.org/library
**Speed of Service Execution**

Table 5 describes the suggested KPIs for speed of service execution.

Table 5. Suggested key performance indicators (KPIs) for speed of service execution.

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>Percentage conformance of the speed of execution of a service against published metrics.</td>
</tr>
<tr>
<td>KPI Justification</td>
<td>It is important for the cloud service subscriber (and provider) to establish a baseline for the timeliness of service execution. This can then be used to assess whether or not future performance metrics for service execution are in conformance with this baseline.</td>
</tr>
<tr>
<td>KPI Calculation Formula</td>
<td>(Number of milliseconds for atomic service execution × 100) ÷ Published time (milliseconds) for atomic service execution</td>
</tr>
</tbody>
</table>
| Suggested Targets  | • Target = 100%  
                      • Threshold = 110%                                                                                                                           |
| If Within Targets  | Orchestrated services are executing within time periods that meet business needs.                                                                  |
| If Below Targets   | The services provided by the cloud service provider(s) are taking longer than expected to execute and could, in extreme cases, be placing time-critical business services at risk. |

**Accuracy of Service Execution**

Table 6 describes the suggested KPIs for accuracy of service execution.

Table 6. Suggested key performance indicators (KPIs) for accuracy of service execution.

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>Conformance of the actual service output(s) against published output(s).</td>
</tr>
<tr>
<td>KPI Justification</td>
<td>The cloud service subscriber must consciously and proactively measure conformance of the expected outputs of a service with the actual received outputs. This will ensure ongoing conformance with the original requirements defined by the cloud service subscriber.</td>
</tr>
<tr>
<td>KPI Calculation Formula</td>
<td>Comparison of the actual service output(s) versus the published service output(s) to assess conformance.</td>
</tr>
</tbody>
</table>
| Suggested Targets  | • Target = 100% conformance  
                      • Threshold = Any non-conformance/unexpected service output(s)                                                                                |
| If Within Targets  | Orchestrated services are accurately delivering the outputs as published by the cloud service provider.                                             |
| If Below Targets   | The services provided by the cloud service provider(s) do not conform to their service. Publication as the actual output(s) are non-conformant. The published services are therefore not delivering the required value and can thereby put the business at risk. |
Legal/Regulatory and Corporate Compliance of Service Output(s)

Table 7 describes the suggested KPIs for legal/regulatory compliance of service output(s).

### Table 7. Suggested key performance indicators (KPIs) for legal/regulatory compliance of service output(s).

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>Conformance of the service output(s) with predefined and applicable legal and/or regulatory constraints and requirements.</td>
</tr>
<tr>
<td>KPI Justification</td>
<td>This KPI can be employed by both cloud service subscribers and cloud service providers, but from slightly different viewpoints. From a subscriber viewpoint, it is essential to measure and control conformance of service outputs with all applicable legal, regulatory and corporate constraints. However, from a provider viewpoint, only the applicable legal and regulatory constraints require consideration from a service design and operation perspective.</td>
</tr>
<tr>
<td>KPI Calculation Formula</td>
<td>Comparison of the actual service output(s) with the legal/regulatory constraint and requirements model(s) to assess conformance.</td>
</tr>
</tbody>
</table>
| Suggested Targets   | Target = 100% conformance  
|                     | Threshold = Any non-conformance of service output(s) with legal/regulatory constraints and requirements                                                                                                        |
| If Within Targets   | Orchestrated services are operating in a legal/regulatory conformant mode.                                                                                                                                       |
| If Below Targets    | The services provided by the cloud service provider(s) do not conform to the applicable legal/regulatory constraints and requirements, and hence service use must cease immediately until conformance is established and verified. |

Cost of Service Execution

Table 8 describes the suggested KPIs for cost of service execution.

### Table 8. Suggested key performance indicators (KPIs) for cost of service execution.

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>The percentage conformance of the actual cost of service execution with the published price list.</td>
</tr>
<tr>
<td>KPI Justification</td>
<td>A cloud service subscriber must be able to assess whether they have been invoiced as expected, with no/minimal “unexpected charges” being levied by the cloud service provider. This is also critical to allow an assessment of budget conformance by the cloud service subscriber for subscribed and orchestrated services.</td>
</tr>
<tr>
<td>KPI Calculation Formula</td>
<td>(Actual cost of service execution ÷ Published price for service execution × 100%) ÷ Published price for service execution</td>
</tr>
</tbody>
</table>
| Suggested Targets   | • Target = 100%  
|                     | • Threshold = 110%                                                                                                                                                                                           |
| If Within Targets   | Orchestrated services are executing within the agreed price list parameters and allocated budget.                                                                                                                                                                         |
| If Below Targets    | The costs for service execution are exceeding allocated budget and the published price list, and are hence not delivering value for money.                                                                      |
Cloud Subscriber Satisfaction
Table 9 describes the suggested KPIs for rating cloud subscriber satisfaction.

Table 9. Suggested key performance indicators (KPIs) for rating cloud subscriber satisfaction.

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>Average cloud subscriber satisfaction survey score.</td>
</tr>
</tbody>
</table>
| KPI Justification | The cloud service provider should assess and ensure that the subscribers of their published services are satisfied with specific parameters of the services offered, such as the following:  
  • The variety of services offered  
  • The agility and flexibility, where possible, for service customization and orchestration  
  • The speed and accuracy of service execution  
  • The transparency of service charging  
  The cloud service provider should regularly (for example, monthly or quarterly) formally request feedback from their subscribers.  
  Note: The cloud service subscriber should provide formal feedback to ensure any issues are addressed and also to formally acknowledge, even to them, that they are happy or dissatisfied with the subscribed services. |
| KPI Calculation Formula | Average cloud subscriber satisfaction survey score. |
| Suggested Targets |  
  • Target = 10.0  
  • Threshold = 9.0  
  Note: Assumes a 10 point scale with 10 = high and 1 = low |
| If Within Targets | Orchestrated services are effectively meeting the stated needs of the cloud service subscriber. |
| If Below Targets | The services provided by the cloud service provider(s) are viewed as problematic and results in a lack of confidence, by the cloud service subscriber, in their capabilities. |

Service Execution Incident Business Impact Ratio
Table 10 describes the suggested KPIs for service execution incident business impact ratio.

Table 10. Suggested key performance indicators (KPIs) for service execution incident business impact ratio.

<table>
<thead>
<tr>
<th>KPI Parameter</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>KPI Statement</td>
<td>Incident business impact ratio</td>
</tr>
<tr>
<td>KPI Justification</td>
<td>While each individual or orchestrated service may have their specific assigned service-level agreements (SLAs), it is difficult for a subscriber to cumulatively assess the impact to the business of unresolved incidents against all subscribed services. Therefore, this KPI hence focuses on a viewpoint of all subscribed services and their operational status, as opposed to the SLA conformance profile of individual subscribed services.</td>
</tr>
<tr>
<td>KPI Calculation Formula</td>
<td>Number of incidents resolved ÷ Total number of incidents reported to service desk × 100%</td>
</tr>
<tr>
<td>Suggested Targets</td>
<td></td>
</tr>
</tbody>
</table>
  • Target = 100%  
  • Threshold = 90% |
| If Within Targets | Orchestrated services are operating without incident and therefore meeting business needs. |
| If Below Targets | The services provided by the cloud service provider(s) are providing little value (or lack of investment in monitoring tools) and, in extreme cases, could be putting business revenue (of cloud service subscriber) and reputation at risk. |
ORCHESTRATION LIFECYCLE OVERVIEW

This section covers the basic lifecycle of an IaaS system whereby the intent is not to provide a fully defined orchestration system; rather the purpose is to communicate how the subscriber interacts with the provider to identify and consume services. This section views key phases and activities of the service orchestration lifecycle as well as the logical architecture and includes services and activities performed within each layer: user interface, business services, and technical services.

Service orchestration describes the automated sequencing, coordination, and management of the required cloud events, systems, and services required to provision, manage, report, execute, and eventually terminate the cloud services. Cloud service delivery entails workflows that connect various technical and business domains in order to align the business request with the applications, data, and infrastructure. The process includes defined policies and service levels and uses automated workflows to provision, execute changes using change management practices, and terminate services when appropriate. Within service orchestration, the three key high-level phases of the lifecycle process are:

• Initiation
• Run, monitor, change
• Termination

The following sections define in more detail each of these key phases along with process steps specific to each phase.

Initiation

A consumer requires cloud services and browses a provider catalog of services. For a private or hybrid cloud offering a contractual and financial agreement will typically be required. This may be a manual process and in some cases may have been automated. For a public cloud offering, the service agreement and payment is often established electronically via the catalog GUI, such as reading a service agreement, checking a box to note acceptance/agreement to the terms, and paying using a credit card or third-party payment system.

The consumer selects the desired service or set of services and composes an order. The subscriber provides the details that are required for the service request. All service request items are collected such as specific business requirements and rules, dependencies, access and privileges, SLAs, and availability requirements.

Once the order is composed the subscriber submits the request. The service provider validates the request for accuracy and completion, reserves the required resources, and then begins the service provisioning.

Run, Monitor, Change

When the service request has been successfully provisioned, the subscriber is notified and is able to access the newly provisioned cloud service. In addition to now having access to the cloud service, the subscriber is also able to monitor dashboards, run reports, request audits, access trending data in order to forecast and plan, and make changes to the running cloud service. The provider performs system administration activities in order to maintain and optimize the cloud operations.

Termination

When the consumer no longer needs the cloud service or the contractual agreement is terminated, the cloud services are deprovisioned. This may be initiated by either the subscriber or provider as agreed upon in the contract. The subscriber receives data for long-term archiving (and any associated audit requirements), according to the contractual agreements. The resources are scrubbed and returned to the available pool for re-allocation. The contract is then concluded and financial billing terminated.

The orchestration lifecycle process flow, as shown in Figure 6, captures service orchestration phases and includes the various stages that occur within each phase of service orchestration.
The service orchestration lifecycle describes how the cloud subscriber interacts with the cloud provider throughout the lifespan of service orchestration of a single service from inception to end of life. The lifecycle is derived from the usage scenarios described earlier in this document and enables the subscriber’s basic needs for identifying and consuming services. Subscribers and providers are both expected to understand the lifecycle and be responsible for their portions of the process. Providers should provide tools and processes to smooth the transition into and through the lifecycle. Subscribers are expected to have clear requirements and access to customer-specific assets to leverage the provider services.

Table 11 further defines the orchestration lifecycle. Each process step is described and the related usage scenario provides more in-depth detail. Please refer to the Usage Scenarios for detailed process steps for each core activity.

Service orchestration enables a cloud consumer to invoke services and changes to those services by means of initiating a number of workflows based on the lifecycle event or phase being triggered.

Orchestration and the associated workflows must comply with corporate and service related rules, as well as support corporate processes and policies. Rules oriented around services and service compilations (for example, HA configuration with redundant storage must be deployed to enable 99.9+% availability), and corporate policies may dictate that a contract must exist between the service subscriber and the service provider before a service catalog may be published or before an actual service can be ordered by a service subscriber. This approach makes common sense, because it lays the foundation for setting expectations and a relationship (whether internal IT-to-business or business-to-business). Therefore, it is identified that there are three key phases in service lifecycles and that orchestration must recognize which phase is relevant to the request and therefore which rules, processes, workflows, and systems must be triggered.

Service orchestration recognizes a series of pre-planned and wired processes and triggers associated with catalog-based services, available activities for those services, and based on rules and processes associated with those activities. As a request is compiled by the service consumer, a package is built up of these activities, workflows, rules, and processes. This determines the specific process flow for that orchestration package as represented in the previous graphic.
### Table 11. Orchestration lifecycle process steps.

<table>
<thead>
<tr>
<th>Process Step</th>
<th>Description</th>
<th>Usage Scenario Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Order Cloud Services</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Account Registration</td>
<td>Agreements including subscriber-provider contract and financial terms are put in place and are dependent on the type of provider offering (public or private). Contractual part of interaction may be manual depending on the service provider and corporate procurement policies. The subscriber accounts and appropriate security access are provisioned.</td>
<td></td>
</tr>
<tr>
<td>Compose Service</td>
<td>The subscriber composes a service order potentially from one or multiple cloud providers. Subscriber provides details and requirements for the selected service request, which may include items such as SLAs, location, dependencies, performance requirements, availability requirements, data classification, security controls, service continuity requirements (RTO, RPO, RCO), and backup requirements. Alternatively, the subscriber may select and open an expired service request instead of composing a new service request. This request may then be revised and submitted.</td>
<td>1. Compose service 6. Reopen expired request</td>
</tr>
<tr>
<td>Validate/Submit Order</td>
<td>The cloud subscriber submits a provisioning request with the provider. The provider validates the request, triggers the reserve resources for service, and submits the order, which in turn initiates the deployment of the service (see Usage Scenario 4 - Deploy Service)</td>
<td>2. Submit provisioning request</td>
</tr>
<tr>
<td>Reserve</td>
<td>The subscriber has composed and submitted a service request. The provider is notified that a service request order is pending and guarantees that the resources needed by the subscriber will be available within the reservation time frame.</td>
<td>3. Reserve resources for service</td>
</tr>
<tr>
<td>Deploy</td>
<td>The cloud provider releases the request and the service provisioning begins. When the request is fulfilled, the subscriber is notified that the servers are available. The cloud subscriber may then make any needed configuration changes and load any applications and data.</td>
<td>4. Deploy service</td>
</tr>
<tr>
<td>Track Request</td>
<td>A service request has been submitted and is being deployed, and the subscriber is able to track the progress of the service request through the provisioning process.</td>
<td>5. Track status and manage deployment</td>
</tr>
<tr>
<td><strong>Run, Monitor, Change</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>The cloud service is started using the appropriate starting order as required. If a test scenario is included in this service, the defined test scenario is executed.</td>
<td>7. Start dependencies</td>
</tr>
<tr>
<td>Stop</td>
<td>The cloud service is stopped and reserved resources are disengaged. The servers are no longer available to subscriber end users and typically billing stops.</td>
<td>8. Stop dependencies</td>
</tr>
<tr>
<td>Suspend</td>
<td>The cloud service is suspended and check-pointing data written if applicable. Resources stay reserved for the suspended service and billing record updated.</td>
<td>9. Suspend cloud service</td>
</tr>
<tr>
<td>Resume</td>
<td>Cloud service is resumed and resources become available again.</td>
<td>10. Resume cloud service</td>
</tr>
<tr>
<td>Monitor</td>
<td>The subscriber monitors the running service for health, performance, availability, capacity, and/or event conditions. Provider notifies subscriber of urgent events and activity.</td>
<td>11. System monitoring, alerting, and data collection</td>
</tr>
<tr>
<td>System Administration</td>
<td>Provider performs system administration activities and remediation. Detected events are addressed and reported acted upon, often with automated scripts.</td>
<td>12. System administration and remediation</td>
</tr>
<tr>
<td>Report</td>
<td>Scheduled and ad hoc reports are generated and published. Subscriber has access to SLAs, billing data, trend analysis, and other reports as defined by the service and/or contract.</td>
<td>13. Reporting</td>
</tr>
<tr>
<td>Forecast</td>
<td>The cloud provider produces a capacity forecast for the subscriber, based on historical usage trends and expected new demand defined by the subscriber. Collected data enables subscriber to predict/review service dimensions, capacity planning, capital expenditures, operational costs, and potential problems.</td>
<td>14. Capacity planning</td>
</tr>
<tr>
<td>Audit</td>
<td>An audit may be initiated at any time during the service lifecycle by the cloud subscriber or provider in response to legal or corporate requirements.</td>
<td>15. Auditing</td>
</tr>
<tr>
<td>Changes/Updates</td>
<td>Make changes to the running service based on new requirements such as the maintenance of applications and data, scaling demand, or other business needs.</td>
<td>16. Change a deployed service instance 17. Autoscale within a cloud</td>
</tr>
<tr>
<td><strong>End/Delete Cloud Services</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comply</td>
<td>The subscriber and provider ensure regulatory requirements associated with service termination are met.</td>
<td>18. Comply to regulatory requirements</td>
</tr>
<tr>
<td>Terminate</td>
<td>The subscriber or provider effects a service termination, which results in the return of compute resources to the pool of available capacity.</td>
<td>19. Service and data termination and deletion</td>
</tr>
</tbody>
</table>
Standard processes are defined by the cloud service provider, and the rules behind the service catalog must align to those processes. These processes must enable compliance as defined by the relevant ODCA Usage Model: Service Catalog applicable to that scenario, security and technology deployment aspects as described in the ODCA Master Usage Model: Compute Infrastructure as a Service (IaaS), and service compilation based on the defined SLAs and services, together with the applicable commercial framework behind those services.

The service provider must understand and define the process and associated workflow interaction for each of the lifecycle activities in advance to support his standard service catalog, and the service consumer must recognize the service lifecycle phase that they are addressing and the expected interactions and triggers associated with their responsibilities in each phase. These predefined processes are defined per the service provider’s corporation and are usually based on the following:

- Compliance requirements at both the country and corporate level, such as applicable legislation, privacy, and rules
- Commercial requirements, such as contract structure and wording
- Security structure, such as Internet or corporate network-based security systems and tools
- Operational environment structure, such as infrastructure location, and service team structures

**USAGE REQUIREMENTS**

The features below are derived from the usage scenarios in the Usage Scenarios, and are aligned with the ODCA Usage Model: Standard Units of Measure. For this usage model, it is not intended that all of the features in a given column must be supported as a group. In practice, a given cloud service provider solution (including the original service provider or through a broker) will combine different service levels for different elements. For example, all Bronze level performance feature requirements must first be met before combining features from any other performance level, and Gold level security features could be combined with Bronze level service catalog features.

A view of the requirements that are important to service orchestration is by means of a MoSCoW prioritization table, which enables stakeholders to understand the importance of each requirement in relation to one another. MoSCoW ratings are added to the right side of Table 12.

As documented in A Guide to the Business Analysis Body of Knowledge, version 2.0, section 6.1.5.2, the MoSCoW categories are as follows:

- **MUST** - Describes a requirement that must be satisfied in the final solution for the solution to be considered a success.
- **SHOULD** - Represents a high-priority item that should be included in the solution if it is possible. This is often a critical requirement but one that can be satisfied in other ways if strictly necessary.
- **COULD** - Describes a requirement that is considered desirable but not necessary. This will be included if time and resources permit.
- **WON'T** - Represents a requirement that stakeholders have agreed will not be implemented in a given release, but may be considered for the future.

As a principle, all requirements are expected to be multi-vendor and open. Key requirements need to be met across consumers and providers.

The general assurance requirements are well documented in the ODCA Usage Model: Provider Assurance. There are some additional requirements specific to service orchestration, but within the same framework of reference, as described in Table 12.
Table 12. General assurance requirements specific to service orchestration.

<table>
<thead>
<tr>
<th>Sub-Category</th>
<th>Bronze</th>
<th>Silver</th>
<th>Gold</th>
<th>Platinum</th>
<th>MoSCoW Provider</th>
<th>MoSCoW Subscriber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orchestration Process and Workflow</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Orchestration Process</td>
<td>Technical process only</td>
<td>Technical and commercial process for requesting, ordering, deploying, and ending cloud services</td>
<td>Technical, commercial, and service process for requesting, ordering, deploying, and ending cloud services</td>
<td>Technical, commercial, and service process for requesting, ordering, deploying, and ending cloud services</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Workflow for Orchestration Cloud Services</td>
<td>Includes a technical workflow for configuring, ordering, deploying, and ending cloud services</td>
<td>Includes a technical and commercial workflow for configuring, ordering, deploying, changing, and ending cloud services, including procurement stages, authorizations, and technical interactions</td>
<td>Includes a technical and commercial workflow for configuring, ordering, deploying, changing, and ending cloud services, including procurement stages, authorizations, and technical interactions</td>
<td>Includes a technical and commercial workflow for configuring, ordering, deploying, changing, and ending cloud services, including procurement stages, authorizations, and technical interactions</td>
<td>M</td>
<td>S</td>
</tr>
<tr>
<td>Service Interface Availability</td>
<td>Up to 4 hours non-availability per month</td>
<td>Up to 1 hour non-availability per month</td>
<td>Up to 10 minutes non-availability per month</td>
<td>Always on</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>Service Interface Performance</td>
<td>Up to 4 hours transaction response time per transaction</td>
<td>Up to 1 hour transaction response time per transaction</td>
<td>Up to 10 minutes transaction response time per transaction</td>
<td>Immediate response per transaction</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>Documented Usage Scenarios</td>
<td>No documented use cases</td>
<td>Availability-focused usage scenarios for cloud services</td>
<td>Have a consistent documented set of usage scenarios for cloud services, to which organizational processes and workflows align</td>
<td>Have a consistent documented set of usage scenarios for cloud services, to which organizational processes and workflows align</td>
<td>M</td>
<td>C</td>
</tr>
<tr>
<td>Orchestration Tooling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>API for Consumer Access/</td>
<td>Infrastructure-only APIs - Programmatic web services in cloud provider's choice of standard</td>
<td>Infrastructure and orchestration system APIs – Programmatic web services in cloud provider's choice of standard</td>
<td>Enables electronic integration between systems and consumers/providers - programmatic web services in recognized industry standard, consistent with ODCA concepts (North usually to consumer, South usually to infrastructure, East and West usually to other clouds/cloud services)</td>
<td>Enables electronic integration between systems and consumers/providers - programmatic web services in recognized industry standard, consistent with ODCA concepts (North usually to consumer, South usually to infrastructure, East and West usually to other clouds/cloud services)</td>
<td>M</td>
<td>S</td>
</tr>
<tr>
<td>Web Services Interface Standard</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GUI</td>
<td>Manual interface via human interaction</td>
<td>Basic service ordering interface</td>
<td>Provides a common interface for managing cloud services and activities</td>
<td>Provides a common interface for managing cloud services and activities</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>Orchestration Tooling</td>
<td>Basic consumer interface only</td>
<td>Enterprise (robustly designed cloud-wide tooling) equivalent tooling for technical orchestration</td>
<td>Tooling to manage commercial and technical phases of orchestration</td>
<td>Secure tooling to manage commercial and technical phases of orchestration</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>Automation of Service Request</td>
<td>No automation</td>
<td>Partial automation with some human intervention stages</td>
<td>Full automation with all options and integration predefined</td>
<td>Full automation, in real time</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reporting</td>
<td>Ad hoc and manually provided reports</td>
<td>Basic standard electronic reports</td>
<td>Reporting interface to display real-time and ad hoc information such as service status, consumption records, and logs</td>
<td>Reporting interface to display real-time and ad hoc information such as service status, consumption records, and logs</td>
<td>M</td>
<td>S</td>
</tr>
<tr>
<td>Sub-Category</td>
<td>Bronze</td>
<td>Silver</td>
<td>Gold</td>
<td>Platinum</td>
<td>MoSCoW Provider</td>
<td>MoSCoW Subscriber</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>-----------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Security Integration</td>
<td>Ad hoc integration per system</td>
<td>Limited integration, with multiple authentication points</td>
<td>Defined architecture for extending consumers security domain and requirements into the cloud provider’s environment, and how that is managed; also governing data protection, encryption, and access rights</td>
<td>Defined architecture for extending consumers security domain and requirements into the cloud provider’s environment, and how that is managed; also governing data protection, encryption, and access rights</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Security Framework</td>
<td>As per ODCA Usage Models: Security Provider Assurance, Identity Management Interoperability Guide, Cloud Based Identity Provisioning; Single Sign-On Authentication</td>
<td>Predefined set of parameters and characteristics to define and communicate available services and elements electronically, to enable brokering and matching against consumer requirements</td>
<td>Predefined set of parameters and characteristics to define and communicate available services and elements electronically, to enable brokering and matching against consumer requirements</td>
<td>Predefined set of parameters and characteristics to define and communicate available services and elements electronically, to enable brokering and matching against consumer requirements</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Service Catalog</td>
<td></td>
<td></td>
<td></td>
<td>Predefined set of parameters to enable communication of consumer requirements in a standardized format, and then enables brokering against available options from the cloud provider</td>
<td>Predefined set of parameters to enable communication of consumer requirements in a standardized format, and then enables brokering against available options from the cloud provider</td>
<td>S</td>
</tr>
<tr>
<td>Defined Service Requirements</td>
<td>Infrastructure-focused definitions, based on cost and performance</td>
<td>Standards-based framework for requirements definition in place</td>
<td>Predefined set of parameters to enable communication of consumer requirements in a standardized format, and then enables brokering against available options from the cloud provider</td>
<td>Predefined set of parameters to enable communication of consumer requirements in a standardized format, and then enables brokering against available options from the cloud provider</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Defined Offering Characteristics</td>
<td>Predefined set of parameters and characteristics available for Infrastructure</td>
<td>Predefined set of parameters and characteristics for determining the management aspects of a service with respect to placement, compliance, and risk dimensions</td>
<td>Predefined set of compliance and risk characteristics for determining the management aspects of a service with respect to placement, compliance, and risk dimensions</td>
<td>Predefined set of compliance and risk characteristics for determining the management aspects of a service with respect to placement, compliance, and risk dimensions</td>
<td>S</td>
<td>M</td>
</tr>
<tr>
<td>Data and Application Categorization</td>
<td>Public and private requirements identified for applications</td>
<td>Disaster recovery requirements identified</td>
<td>Predefined set of policies to enable the management of data according to organizational requirements</td>
<td>Predefined set of policies to enable the management of data according to organizational requirements</td>
<td>S</td>
<td>M</td>
</tr>
<tr>
<td>Service Catalog Updates</td>
<td>Updated Monthly</td>
<td>Updated monthly and on request</td>
<td>Updated daily</td>
<td>Updated immediately</td>
<td>S</td>
<td>C</td>
</tr>
<tr>
<td>Defined Service Lifecycle</td>
<td></td>
<td></td>
<td></td>
<td>Predefined set of policies to enable the management of data according to organizational requirements</td>
<td>Predefined set of policies to enable the management of data according to organizational requirements</td>
<td>S</td>
</tr>
<tr>
<td>Defined Data Policies and Protection</td>
<td>Global single standard recommended</td>
<td>Core and non-core applications identified, with differences, from a business perspective</td>
<td>Predefined set of policies to enable the management of data according to organizational requirements</td>
<td>Predefined set of policies to enable the management of data according to organizational requirements</td>
<td>S</td>
<td>M</td>
</tr>
<tr>
<td>Governance Framework</td>
<td></td>
<td></td>
<td></td>
<td>List of requirements specifying data, reports, and interactions expected during each step of the service’s usage scenarios, including which systems or interfaces must provide and receive information</td>
<td>List of requirements specifying data, reports, and interactions expected during each step of the service’s usage scenarios, including which systems or interfaces must provide and receive information</td>
<td>S</td>
</tr>
<tr>
<td>Predefined Requirements Enabling the Usage Scenarios</td>
<td>Basic usage scenarios for Infrastructure identified, ad hoc</td>
<td>Systems, steps, and processes mapped against usage scenarios</td>
<td>List of requirements specifying data, reports, and interactions expected during each step of the service’s usage scenarios, including which systems or interfaces must provide and receive information</td>
<td>List of requirements specifying data, reports, and interactions expected during each step of the service’s usage scenarios, including which systems or interfaces must provide and receive information</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>Service Reporting</td>
<td>Monthly</td>
<td>Weekly</td>
<td>Real-time</td>
<td>Real-time</td>
<td>S</td>
<td>C</td>
</tr>
</tbody>
</table>
RFP REQUIREMENTS — SERVICE PROVIDERS

The following are requirements that the Alliance suggests should be included in requests for proposal to cloud providers to ensure that proposed services support service orchestration. Click here for an online assistant, Proposal Engine Assistant Tool (PEAT),33 to help you detail your RFP requirements.

• ODCA Principle Requirement
  – Service is open and is standards-based. Describe how the service meets this principle and any limitations toward the ODCA principle.

• ODCA Service Orchestration Usage Model Rev 2.0
  – A cloud orchestration GUI portal is available that enables the full orchestration of the cloud service and is available at 99.99%. The GUI includes service catalog navigation, simplified ordering, and triggering deployment of services.
  – A CLI is available that enables the full orchestration of the cloud service and is available at 99.99%. The CLI includes service catalog navigation, simplified ordering, and triggering deployment of services.
  – An API is available that enables the full orchestration of the cloud service and is available at 99.99%. The API includes service catalog navigation, simplified ordering, and triggering deployment of services. APIs conform to a REST architectural model.
  – Service lifecycle phases are fully documented and available online, with defined sub-steps, actions, and success/failure conditions. The documentation must include the Automated Service Provisioning Process, from beginning to end, and indicate any manual steps involved.
  – A downtime and maintenance schedule is available on-demand, indicating which functions and service elements are maintained online and which require downtime.
  – Describe your handling of potential availability issues such as significant cloud computing outage, high network load, or insufficient bandwidth access.
  – Cloud orchestration portal GUI response time is <2 seconds.
  – Committed service request completion times are provided for cloud service requests as follows: (1) new VM provisioning, (2) add, move, or change and (3) service termination and deletion.
  – An online graphic representation shows actual service deployment stages against the whole process, compared to committed times for that phase in the orchestration process, as per the master services agreement.
  – Define which open standards for external interfaces are employed, if there are any proprietary standards, and what cost/licensing is required for using them.
  – Provide a draft contract for cloud services showing the availability of standard contract clauses for service orchestration, defining expectations, obligations, dependencies, rules, and processes.
  – Conform the compliance of service orchestration process to legislation (country and corporate) and indicate what legislation and standards are certified.
  – Describe how the billing/invoicing model promotes predictability for customers. Please provide an example invoice (without revealing customers).
  – Define what kind of trouble ticketing system is used and whether it is visible from the subscriber’s site.
  – Describe the level of automatic alerting that can be provided to the subscriber support staff in the event of failure, degraded service, or exceeded planned utilization.
  – Define what controls are in place for administrative access, both internal to your company and for administrative access from subscriber clients. Please include discussion of administrator controls over provisioning.
  – Describe your experience in weaving together multiple different cloud computing services offered by you, if any, or by other vendors.
  – Define the logging and auditing facilities available for the orchestration API, CLI, and GUI, for purposes of auditing and tracing of transactions, and for what period the data is retained.
  – Describe the security and encryption applicable within the orchestration API.
  – Service orchestration programming examples must be made available and in commonly used languages. Outline what client libraries (also known as “bindings” or “proxies”) are available for various commonly used languages to accelerate adoption.
  – Web services provide mechanisms to support async call back (that is, publish results to message queue where consumer is listening, email notification, and so on) for long-running tasks. The task should be identified by a globally unique identifier. Subscriber can query for status/progress of the task using this ID.
  – Web services are compartmentalized and capable of scaling out and back, supporting elasticity at massive scale.

33 ODCA Proposal Engine Assistant Tool (PEAT). www.opendatacenteralliance.org/ourwork/proposalengineassistant
Web services are designed to be accessed through meaningful and easily remembered vanity names in combination with global load balancers with appropriate routing policies (such as proximity or round robin).

Web services are designed for failure, anticipating infrastructure failures, and maintaining uptime across multiple availability zones.

Maintenance and ongoing development of the services interfaces are non-intrusive (backward compatible), enabling sustained operations without impact or requirement for downtime. Incompatible changes are addressed through API versioning so that the old and new versions are available simultaneously.

Is a service catalog available electronically, and does it describe the following service element characteristics based on (an) industry standard format/s: Technology functionality, capacity available, performance thresholds, service level capabilities, provisioning times, security, carbon footprint, commercial elements including price/contract terms.

Please provide a Future Roadmap and Timeline Planning register of feature and function updates, for the cloud services and component elements provided as part of the service.

**SUMMARY OF INDUSTRY ACTIONS REQUIRED**

In order to provide guidance in the creation and deployment of solutions that are open, multi-vendor, and interoperable, the ODCA has identified specific areas where we suggest there should open specifications, formal or de facto standards, or common intellectual property-free implementations. Where the ODCA has a specific recommendation on the specification, standard or open implementation, it is called out in this usage model. In other cases, we plan to work with the industry to evaluate and recommend specifications in future releases of this document.

The following are industry actions required to refine this usage model:

- Define a service orchestration standard API between service providers and service receivers.
- Define open standards for advertisement of cloud services to cloud brokers.
- Define the key integration points between cloud service providers and service consumers, and the minimum expected functionality from each side, to enable open standards based automation between them.
- Version 1.0 of this master usage model has focused on requirements for a single cloud subscriber engaged with a single cloud provider. While acknowledging the role of cloud brokers, cloud federation, and cloud marketplaces, there has been little elaboration on how they impact IaaS requirements. Future versions of the ODCA Master Usage Model: IaaS should address those issues.
- Define standards for advertising IaaS catalog entries that include technical, service, and commercial dimensions, in order that PaaS and SaaS software may select them based on predefined preferences.
- Define standards for encapsulating catalog entries for advertisement, all the way from IaaS through PaaS, to SaaS levels.

**OTHER SOURCES**

- DMTF’s Cloud Management for Communications Service Providers www.dmtf.org/sites/default/files/standards/documents/DSP2029%20_1.0.0a.pdf
- ODCA Proposal Engine Assistant Tool (PEAT) www.opendatacenteralliance.org/ourwork/proposalengineassistant

See www.opendatacenteralliance.org/library