INTERNATIONAL STANDARD

ISO/IEC 19941

First edition
2017-12

Information technology — Cloud computing — Interoperability and portability

Technologies de l'information — Informatique en nuage — Interopérabilité et portabilité

Reference number
ISO/IEC 19941:2017(E)

COPYRIGHT PROTECTED DOCUMENT

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword ..... **v**

Introduction ..... **vi**

1 Scope **1**

2 Normative references **1**

3 Terms and definitions **1**
3.1 Interoperability terms 1
3.2 Data portability terms 2
3.3 Application portability terms 2

4 Abbreviated terms **3**

5 Overview of cloud computing interoperability and portability **4**
5.1 Description of cloud computing interoperability and portability 4
5.1.1 General 4
5.1.2 Considerations for cloud interoperability 5
5.1.3 Considerations for portability in a cloud computing environment 6
5.1.4 Relationship between cloud interoperability and cloud portability 9
5.2 Cloud interoperability and portability facet models 9
5.2.1 Cloud interoperability facet model 9
5.2.2 Cloud data portability facet model 13
5.2.3 Cloud application portability facet model 16
5.3 Key challenges related to interoperability and portability in cloud computing 18
5.3.1 General 18
5.3.2 Security 18
5.3.3 Identity and Access Management (IdAM) 20
5.3.4 Security during migration 21
5.3.5 Dynamic migration 21
5.3.6 Interfaces, APIs and interoperability 21
5.3.7 Open source 22

6 Interoperability and portability considerations related to cloud capabilities types **22**
6.1 General 22
6.2 Functional components of interoperability 27
6.3 Functional components of data portability 28
6.4 Functional components of application portability 28
6.4.1 General 28
6.4.2 Functional views based on capabilities types 31

7 Cloud interoperability **36**
7.1 Cloud interoperability types 36
7.1.1 General 36
7.1.2 Transport interoperability 38
7.1.3 Syntactic interoperability 39
7.1.4 Semantic data interoperability 39
7.1.5 Behavioural interoperability 39
7.1.6 Policy interoperability 40
7.1.7 Interoperability with connected devices consuming cloud services of application capabilities type 41

8 Cloud data portability **42**
8.1 Cloud data portability types 42
8.2 Data syntactic portability 42
8.2.1 General 42
8.2.2 Data syntactic portability for infrastructure capabilities type cloud services 43
8.2.3 Data syntactic portability for platform capabilities type cloud services 43
8.2.4 Data syntactic portability for application capabilities type cloud services 44

8.3 Data semantic portability 45
8.3.1 General 45
8.3.2 Data semantic portability for infrastructure capabilities type cloud services 45
8.3.3 Semantic data portability for platform capabilities type cloud services 45
8.3.4 Data semantic portability for application capabilities type cloud services 45
8.4 Data policy portability 47
8.5 Considerations for cloud data portability of "cloud service derived data" 47

9 Cloud application portability 49
9.1 Cloud application portability types 49
9.2 Considerations for cloud application portability 50
9.3 Application portability for infrastructure capabilities type cloud services 53
9.3.1 Application portability from non-cloud to cloud service 53
9.3.2 Application portability from a cloud service to another cloud service 55
9.4 Application portability for platform capabilities type cloud services 57
9.4.1 Application portability from non-cloud to cloud service deployments 57
9.4.2 Application portability from one cloud service to another cloud service 60
9.5 Application portability for application capabilities type cloud service 62
9.6 Application portability: Policy facet 62
9.6.1 General 62
9.6.2 Law and regulations 62
9.6.3 Contracts and licenses 63
9.6.4 Organizational policies 63

Bibliography 64

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 38, *Cloud Computing and Distributed Platforms*.

Introduction

This document is intended to establish a common understanding of cloud computing interoperability and portability. In particular, it is of interest to cloud stakeholders focusing on cloud service agreements concerning interoperability or portability between cloud services.

Cloud computing is defined as a paradigm for enabling network access to a scalable and elastic pool of shareable physical or virtual resources with self-service provisioning and administration on-demand. ISO/IEC 17788 and ISO/IEC 17789 provide a starting point for understanding of different types of interoperability and portability, relationships with activities and roles and cloud capabilities types. Interoperability, data portability and application portability are essential to the use of cloud services. The goal of interoperability is to enable the interaction between non-cloud and cloud services, as well as between cloud services, in addition to enabling composition of new services from multiple services. The goal of portability is to enable cloud service customers (CSCs) to move their data or applications between non-cloud and one or more cloud services and between cloud services. The benefits of interoperability include lower costs of integration and increasing the value of services through enrichment or new functionality provided by composing cloud services. The benefits of portability include greater efficiency by lowering the costs of migration. Both interoperability and portability offer more choices to CSCs by limiting the effects of being locked in to any cloud service or cloud service provider (CSP). While there is no disagreement that interoperability and portability are advantages to cloud computing, there is no single way of handling either capability. Declaring interoperability or portability without doing a detailed analysis of what specifically is to be ported or is to be made interoperable is meaningless and does not lead to cloud solutions that meet the CSC's and CSP's business goals, which has led to significant and on-going confusion in the industry and needs to be resolved.

Interoperability is the ability of two or more systems or applications to exchange information and to mutually use the information that has been exchanged. In the context of cloud computing, interoperability should be viewed as the capability of public cloud services, private cloud services and other cloud service customer systems to understand each other's interfaces, configuration, forms of authentication and authorization, etc. in order to cooperate and work with each other.

Interoperability is a complex subject in the context of cloud computing because of the number of interactions involved and the potential variations for each interaction. While interoperability and standards add significant value and are advantageous to cloud computing, there are no comprehensive solutions. Many existing IT standards contribute to enabling interoperability between CSC applications and cloud services and between cloud services themselves. Using standards can be one way to build interoperable cloud services. Other techniques such as well-documented API specifications can also help.

Cloud computing services that enable portability using defined policies, standards or documented formats can ensure that CSCs are able to get their data into or out of cloud services in a reasonably easy and cost-effective manner, as this allows CSCs to move to a cloud service of another CSP and also to drive integration of heterogeneous cloud services.

As presented in ISO/IEC 17788, portability is the ability of a CSC to move their data or their applications between two different cloud services at a low cost and with minimal disruption. Portability is significant in cloud computing since CSCs are interested in avoiding lock-in when they choose to use cloud services. Therefore, in the context of cloud computing, portability can have multiple aspects depending on what is being ported (moved) and which cloud services are involved. For portability, there are no specific requirements for the source and target systems to be directly connected.

Portability in a cloud computing environment is not a binary concept. It would be a mistake to think of cloud services and the associated cloud applications and data as being either 100% portable or not portable at all. Almost all applications running in a cloud service can be ported to another cloud service offering equivalent capabilities if enough resources are invested. The critical considerations for portability discussions are the porting cost, the risks associated with the porting and how to control the costs and risks compared to the expected benefits.

Information technology — Cloud computing — Interoperability and portability

1 Scope

This document specifies cloud computing interoperability and portability types, the relationship and interactions between these two cross-cutting aspects of cloud computing and common terminology and concepts used to discuss interoperability and portability, particularly relating to cloud services.

This document is related to other standards, namely, ISO/IEC 17788, ISO/IEC 17789, ISO/IEC 19086-1, ISO/IEC 19944, and in particular, references the cross-cutting aspects and components identified in ISO/IEC 17788 and ISO/IEC 17789 respectively.

The goal of this document is to ensure that all parties involved in cloud computing, particularly CSCs, CSPs and cloud service partners (CSNs) acting as cloud service developers, have a common understanding of interoperability and portability for their specific needs. This common understanding helps to achieve interoperability and portability in cloud computing by establishing common terminology and concepts.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at http://www.iso.org/obp

3.1 Interoperability terms

3.1.1
interoperability
ability of two or more systems or applications to exchange information and to mutually use the information that has been exchanged

[SOURCE: ISO/IEC 17788:2014, 3.1.5]

3.1.2
cloud interoperability
ability of a CSC's system to interact with a cloud service or the ability for one cloud service to interact with other cloud services by exchanging information according to a prescribed method to obtain predictable results

Note 1 to entry: Cloud service to cloud service interactions occur through a CSP: inter-cloud provider relationship.

3.1.3
transport interoperability
interoperability (3.1.1) where information exchange uses an established communication infrastructure between the participating systems

3.1.4
syntactic interoperability
interoperability (3.1.1) such that the formats of the exchanged information can be understood by the participating systems

3.1.5
semantic data interoperability
interoperability (3.1.1) so that the meaning of the data model within the context of a subject area is understood by the participating systems

3.1.6
behavioural interoperability
interoperability (3.1.1) so that the actual result of the exchange achieves the expected outcome

3.1.7
policy interoperability
interoperability (3.1.1) while complying with the legal, organizational and policy frameworks applicable to the participating systems

3.2 Data portability terms

3.2.1
data portability
ability to easily transfer data from one system to another without being required to re-enter data

Note 1 to entry: It is the ease of moving the data that is the essence here. This might be achieved by the source system supplying the data in exactly the format that is accepted by the target system. But even if the formats do not match, the transformation between them may be simple and straightforward to achieve with commonly available tools. On the other hand, a process of printing out the data and rekeying it for the target system could not be described as "easy".

[SOURCE: ISO/IEC 17788:2014, 3.2.21]

3.2.2
cloud data portability
data portability (3.2.1) from one cloud service to another cloud service or between a CSC's system and a cloud service

[SOURCE: ISO/IEC 17788:2014, 3.2.6, modified — "or between a CSC's system and a cloud service" has been added.]

3.2.3
data syntactic portability
data portability (3.2.1) using data formats that can be decoded on the target

3.2.4
data semantic portability
data portability (3.2.1) such that the meaning of the data model is understood within the context of a subject area by the target

3.2.5
data policy portability
data portability (3.2.1) while complying with the legal, organizational and policy frameworks applicable to both the source and target

3.3 Application portability terms

3.3.1
application portability
ability to migrate an application from a source system to a target system

3.3.2
cloud application portability
ability to migrate an application from one cloud service to another cloud service or between a CSC's system and a cloud service

[SOURCE: ISO/IEC 17788:2014, 3.2.2, modified — "or between a CSC's system and a cloud service" has been added.]

3.3.3
application syntactic portability
application portability (3.3.1) where the format of the application artefacts can be decoded on the target

3.3.4
application instruction portability
application portability (3.3.1) so that the application's instruction set executes on the target

3.3.5
application metadata portability
application portability (3.3.1) so that the application's metadata is retained and understood on the target

3.3.6
application behaviour portability
application portability (3.3.1) so that execution on the target produces equivalent results to those produced on the source

3.3.7
application policy portability
application portability (3.3.1) while complying with the legal, organizational and policy frameworks applicable to the source and target

4 Abbreviated terms

API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation 1
BPEL	Business Process Execution Language
BPML	Business Process Management Language
CRM	Customer Relationship Management
CSC	Cloud Service Customer
CSN	Cloud Service Partner
CSP	Cloud Service Provider
CSV	Comma-separated values
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
HCM	Human Capital Management
HTTP	Hyper Text Transfer Protocol

IaaS	Infrastructure as a Service
ICT	Information & Communication Technology
IdAM	Identity and Access Management
JSON	JavaScript Object Notation
MIME	Multipurpose Internet Mail Extensions
MQTT	Message Queuing Telemetry Transport
OVF	Open Virtualization Format
OWL	Web Ontology Language
PaaS	Platform as a Service
PII	Personally identifiable information
REST	Representational State Transfer
SaaS	Software as a Service
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
UML	Unified Modeling Language
VM	Virtual Machine
VPN	Virtual Private Network
XML	eXtensible Markup Language

5 Overview of cloud computing interoperability and portability

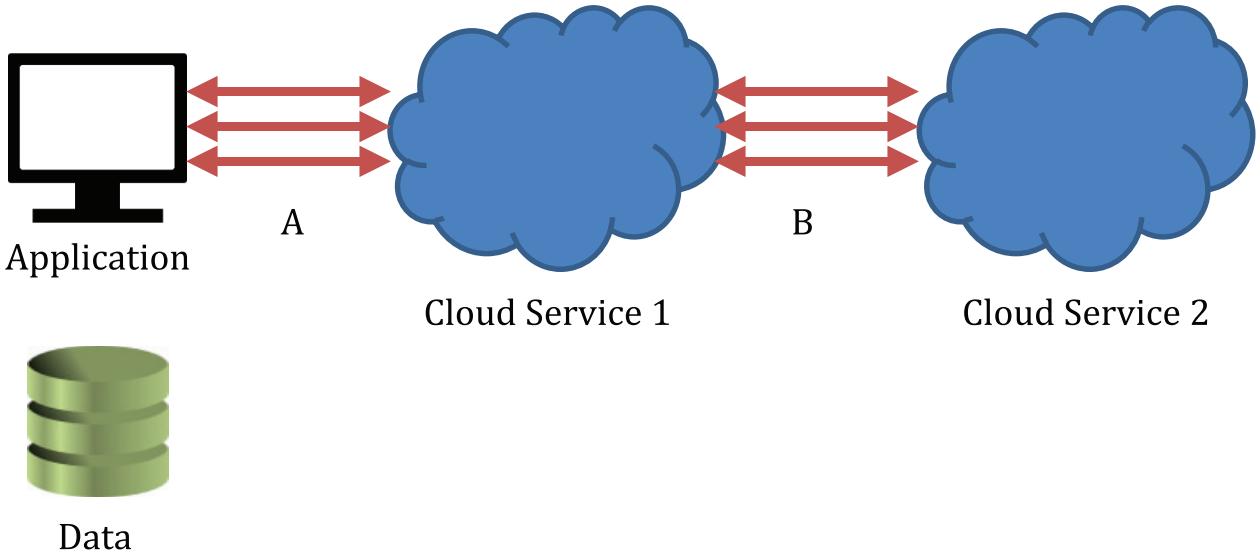
5.1 Description of cloud computing interoperability and portability

5.1.1 General

This clause provides an overview and models (known as "facet models"; refer to 5.2.1, 5.2.2 and 5.2.3 for details) for cloud interoperability, cloud data portability and cloud application portability. There are various perspectives of interoperability and portability that need to be considered. These perspectives are called "facets".

Interoperability and portability in cloud computing involve interactions affected by technological, information and human aspects. Interoperability and portability related challenges are likely to intensify and become more difficult to manage as systems grow more complex and interconnected. In cloud computing environments with internationally interconnected systems, the complexities also include matters of corporate policy, regulation and international law.

5.1.2 Considerations for cloud interoperability



Key

A application to/from cloud service

B cloud service to/from cloud service

Figure 1 — High-level view of cloud interoperability

ISO/IEC 17788 defines interoperability as "*the ability for two or more systems or applications to exchange information and mutually use the information that has been exchanged*". In the context of cloud computing, interoperability is further described as a cross-cutting aspect providing the ability for a cloud service customer system to interact with a cloud service and exchange information according to a prescribed method and obtain predictable results (see ISO/IEC 17788:2014, 6.6). Interoperability also includes the ability for one cloud service to interact with other cloud services (see ISO/IEC 17789:2014, 8.5.5). Figure 1 indicates that cloud interoperability takes place both between a CSC's application and cloud services and also takes place between cloud services. It is also notable that there are typically multiple interfaces involved in both of these cases, as indicated by the multiple arrows.

Note that interoperability in cloud computing is rarely confined to a binary decision of possible or impossible. More often, interoperability is possible subject to implementation costs. A cost/benefit analysis is required to determine whether the resources needed to assure exchange of information in the prescribed method while obtaining predictable results is worthwhile. The ability of systems of a CSC and cloud services as well as multiple cloud services to interoperate with respect to the facets discussed below is more than a matter of investing the resources to assure the exchange of information between the interfaces at either end, since interoperability also requires validation that behavioural and policy facets are respectively compatible. In addition, any changes caused by interoperation requirements may entail additional training for end users, management and operations staff.

There are many considerations when addressing cloud interoperability. These include:

— the ability of a CSC to interact with a cloud service by exchanging information according to a prescribed method obtaining predictable results;

— the ability for a cloud service to work with other cloud services;

— properties needed to facilitate successful interactions between an organization's ICT facilities and a cloud service;

— roles and activities as defined in ISO/IEC 17789;

— cloud capabilities types as defined in ISO/IEC 17788;

— interfaces between different functional components as defined in ISO/IEC 17789:2014, 9.2.

By taking these considerations into account, this document promotes better understanding of the requisites for interoperable cloud services.

5.1.3 Considerations for portability in a cloud computing environment

5.1.3.1 General

This document distinguishes between cloud application portability and cloud data portability. In the context of cloud computing, portability refers to the ability of a CSC to move and suitably adapt their applications and data between the CSC's systems and cloud services, between different cloud deployment models, and between cloud services of different CSPs.

Note that portability in cloud computing is rarely confined to a binary decision of possible or impossible. More often, portability is "possibly subject to switching costs". A cost/benefit analysis is required to determine whether porting applications and/or data is worthwhile. The similarity of the CSC and CSP's systems with respect to the facets described in 5.2.2 and 5.2.3 is therefore more of a matter of lowering the switching cost than of "enabling" portability to take place, since almost any portability is possible if the customer is willing and able to pay for it. Switching concerns are not limited to costs; it also usually involves some risks and usually entail the CSC spending effort and time and perhaps a period of service interruption.

There are many considerations when addressing portability in cloud computing. These include:

— allowing CSCs to migrate applications and data in response to business needs such as faster service, lower cost, greater reliability or disaster recovery needs;

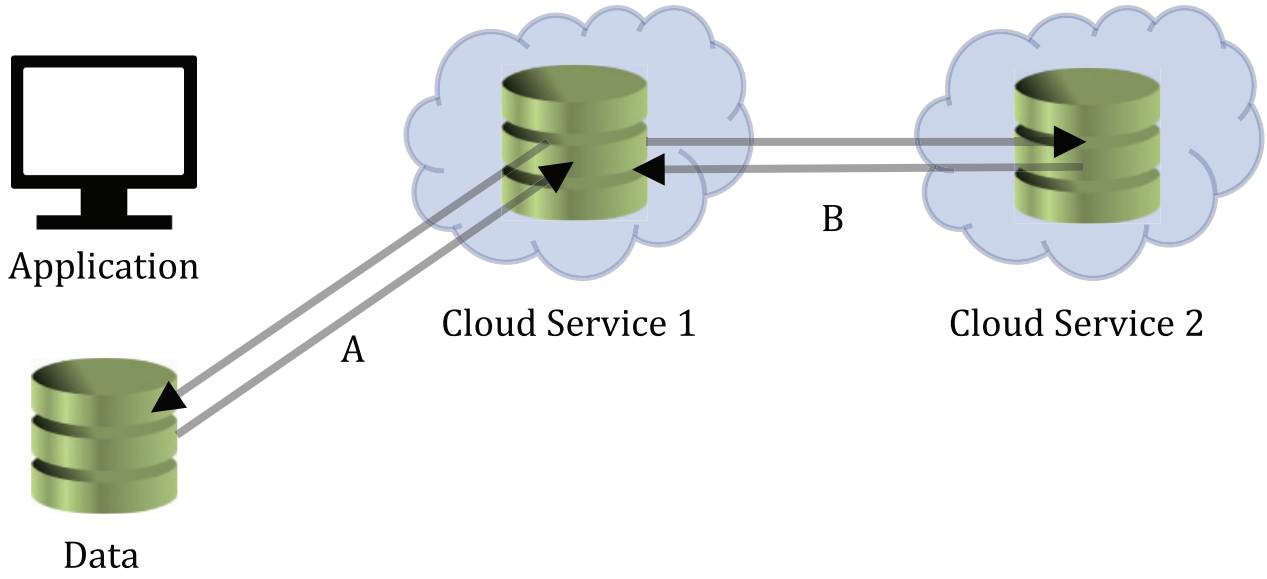
— wider availability of application and data allowing access to a broader market;

— time and effort required for porting both applications and data, however, such overhead may be reduced using common programming languages, standards, tools, frameworks, models, run times and APIs;

— limiting of lock-in situations where the CSC is tied to the cloud services of one CSP.

Portability is an aspect of the more general topic of migration. Other issues related to migration are not considered further in this document.

5.1.3.2 Cloud data portability



Key

A CSC system to/from cloud service

B cloud service to/from cloud service

Figure 2 — High-level view of cloud data portability

Cloud data portability is the ability to transfer data from one cloud service to another cloud service or between a cloud service customer's system and a cloud service. Figure 2 indicates the porting of data between a CSC's system and a cloud service and porting of data from one cloud service to another. The arrows in both directions indicate the potential to port data to and from any of those places.

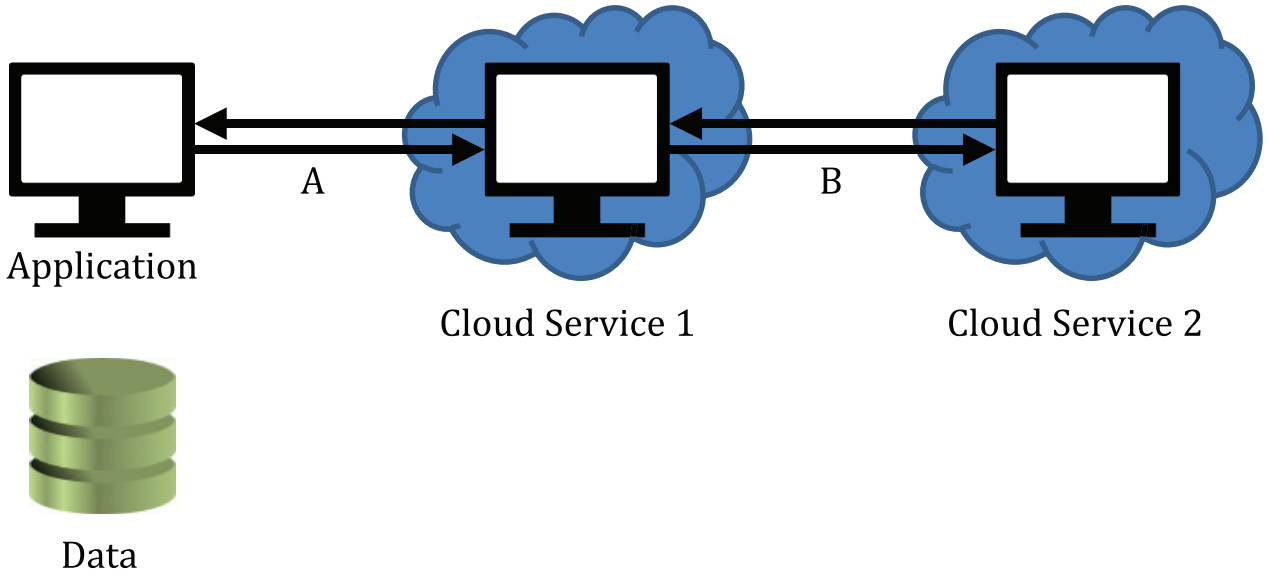
Considerations relating to cloud data portability include:

— retrieval of cloud service customer data. A capability to retrieve cloud service customer data from the source cloud service is needed and a capability to import cloud service customer data into the target cloud service. Cloud data is frequently large enough to tax available bandwidth between systems and data might therefore be moved by the physical movement of physical storage media. In some cases, data is moved electronically;

— syntax of the data. The syntax of the data is ideally the same for the source service and the target service. However, if the syntax does not match, e.g. the source uses JSON syntax but the target uses XML, it may be possible to map the data using commonly available tools;

— semantics of the data. The semantics of data are commonly expressed by an ontology. Compatible ontologies simplify the porting of data between source and target services. If the ontologies are incompatible, additional resources may be applied to detect inconsistences. These inconsistencies may be resolved or fidelity of the data may be reduced to enable the data to be ported.

5.1.3.3 Cloud application portability



Key

A CSC system to/from cloud service

B cloud service to/from cloud service

Figure 3 — High level view of cloud application portability

Cloud application portability is the ability to migrate applications from a CSC's system to a cloud service or from one cloud service to another including migration between instances of cloud deployment models (private, public, community and hybrid). Figure 3 shows the porting of an application between a CSC's system and a cloud service and porting of an application between two cloud services. The arrows in both directions indicate the potential to port applications to and from any of those places.

Considerations relating to cloud application portability include the following.

— Cloud application portability can require the movement of one or more application components that form part of a larger, multi-cloud application. For instance, in addition to application logic, it may be necessary to port and/or reconfigure the cloud application and/or the components upon which it depends, e.g. libraries, databases and web servers. The sequence of virtual machine and/or component start-up may also be important. Portability of complex applications may also require CSPs to share application metadata. This metadata might be acquired by capturing expert knowledge and best practices related to that application's deployment and subsequent management throughout its lifecycle, by automated inspection or discovery or by other means. Common examples of this metadata are details regarding the relationships and dependencies between various application components, requirements such as the acceptable range of component versions, start-up sequence, network and firewall configuration, processing capacity, co-location rules and load balancing requirements.

— Cloud application portability requires that interfaces needed by the application in the source environment are also available in the target environment. These interfaces, for example, might enable the application to use service discovery and communication protocols implemented by the environment, as well as providing access to the environment capabilities that support the application. In some environments, the interfaces may also enable applications to manage the underlying resources. In cases where an application is being ported between two cloud services, the ability of a target cloud service to replicate the environment that the source cloud service has for the application/service or at least create an environment that similarly satisfies the dependencies of the application, is a major consideration.

— The reduced disruption and increased choice enabled by cloud application portability provide CSCs with the capability to mitigate risks. Cloud application portability can facilitate greater business agility by enabling more rapid redeployment of cloud applications and services to alternative or complementary CSPs in response to changing business conditions and technical trends.

— Cloud application portability requires that the identified activities of CSC and CSN and their sub-roles that are supported in the source system are also supported, with acceptable fidelity, by the target system and its components. In practice, different cloud services rarely provide identical capabilities to support all of the activities for all sub-roles. The effort necessary to adjust for these differences and the potential benefits need to be considered. For example, a cloud application implemented on a infrastructure capabilities type cloud service (ISO/IEC 17788:2014, 3.2.25) moved to a different cloud service of the same type might provide identical capabilities to support the activities of the CSC:Cloud service user sub-role deploying and operating the application, but very different capabilities for the CSC:Cloud service administrator sub-role managing the use of the cloud service.

5.1.4 Relationship between cloud interoperability and cloud portability

It is important to understand that portability and interoperability are not synonymous. While interoperability and portability are often discussed in parallel and are related concepts, they are in fact separate concepts without direct dependencies.

The focus of interoperability is the ability to exchange information between a CSC's system and a cloud service or between a cloud service and another cloud service, resulting in the ability to mutually use the information that has been exchanged. A cloud service that is interoperable does not necessarily support portability of applications and/or data.

Portability is the ability to migrate data or applications from one cloud service to another or between a CSC's system and a cloud service. The degree of effectiveness and efficiency of the migration is considered as the ability to execute the application or use the data with as few or no manual changes in the migration process as described in ISO/IEC 17788. The focus of portability is the ease of migration of the data and application. A cloud service that supports portability is not necessarily interoperable.

5.2 Cloud interoperability and portability facet models

5.2.1 Cloud interoperability facet model

5.2.1.1 General

Interoperability is not a simple "yes/no" concept. Interoperability involves a number of elements, starting at the simple exchange of data bytes, facilitating an understanding of the semantics of the exchanged information and also an alignment of the business processes, behaviour and policies on either side of the exchange. It can be that semantic, behavioural and policy interoperability is a significantly bigger challenge than the bits and bytes.

The interoperability facet model described in this document defines five facets within the context of cloud interoperability. These five facets, shown in Figure 4, are transport, syntactic, semantic data, behavioural and policy. This model is derived by combining and abstracting the European Interoperability Framework[13] and the Levels of Conceptual Interoperability Model (LCIM)[14].

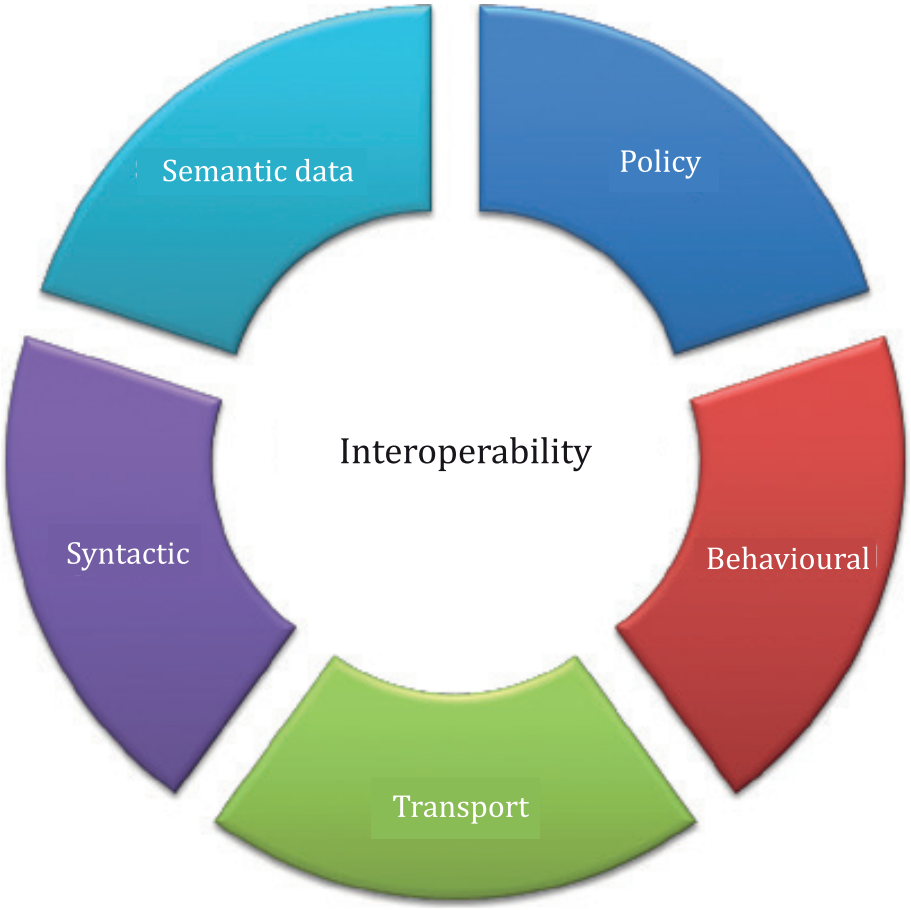


Figure 4 — Facets of cloud interoperability

5.2.1.2 Transport interoperability

Transport interoperability means the commonality of the communication infrastructure established to exchange data between systems. In the context of cloud computing, transport interoperability is the transfer mechanism between various cloud computing components, either between CSC components and CSP components or between CSP components related to different cloud services. Examples include HTTP/S, SOAP, Advanced Message Queuing Protocol (AMQP) and Message Queuing Telemetry Transport (MQTT).

5.2.1.3 Syntactic interoperability

Syntactic interoperability is the ability of two or more systems or services to understand the structure of exchanged information, which is an encoding of the domain concepts as defined by the semantic data facet. Examples of encoding syntaxes include JSON, XML and syntaxes described in ASN.1.

5.2.1.4 Semantic data interoperability

Semantic data interoperability is the ability for the systems exchanging information to understand the meaning of the data model within the context of a subject area. Domain concepts in a cloud computing context are dictated by the type of cloud service offering.

Semantic data interoperability is based on the data models of the information being exchanged at the time of that exchange. At the infrastructure level, this concerns virtual machines (VMs), containers, storage and networking concepts and their management. At the platform level, this concerns applications, the execution and deployment environment and their management. At the application

level, the domain concepts are dictated by the application itself, such as Human Capital Management (HCM), Customer Relationship Management (CRM), Enterprise Resource Planning (ERP), etc.

At the business domain level, semantic data interoperability concerns the ability for discrete domain concepts to be shared and understood between applications, e.g. the concept of "customer" in insurance versus the concept of "patient" in healthcare. Example approaches include the construction of ontologies using, for example, OWL, and the use of semantic query languages like SPARQL.

5.2.1.5 Behavioural interoperability

Behavioural interoperability is where the results of the use of the exchanged information matches the expected outcome. A cloud service, like any other piece of software is designed for a particular purpose or intention. However, its actual use by a customer may have a different intention without violating the other facets of interoperability. For example, the web architecture was originally intended to serve static web pages but over time and without significant architectural redesigns, many different dynamic and interactive models have emerged.

Behavioural interoperability of a cloud service is defined in the service description. The service description includes a declaration of the interface provided by the service, often referred to as an API. The interface declaration describes the service in terms of a set of operations provided by the service and the inputs and outputs for each operation. In terms of the service description, behavioural interoperability requires additional information to be supplied in terms of the expected results of each operation, including elements such as pre-conditions, post-conditions and any sequences of operations that are necessary for successful use of the service.

Behavioural interoperability is defined in terms of:

a) Intended use as described by the CSP. This description is a characterization of the functionality provided by the service.

b) Expected results as defined by the cloud service description include:

   1) pre-conditions as stated by the CSP which need to be fulfilled prior to operation;

   2) post-conditions which reflect the state of the service upon completion of the operation;

   3) orchestration and dependencies with respect to other services required by the CSP to assure correct operation.

The behavioural interoperability facet abstracts from implementation details and describe the behaviour of software components in a representation-independent way.

If the result of an operation against an interface is coherent with the customer expectations for two different cloud services, the cloud services are considered interoperable, assuming identical policies have been applied.

As an example, if the CSC's current order processing system automatically waits for approval of a submitted order by a listed authority, while a new system assumes that such an order is already approved, the behaviour is very different and problems will arise, although both process the same order data otherwise correctly.

5.2.1.6 Policy interoperability

Policy interoperability is defined as the ability of two or more systems to interoperate while complying with the legal, organizational and policy frameworks applicable to the participating systems. This facet concerns governmental laws and regulations, CSP and CSC policy terms and conditions and organizational policies covering the interactions. Governmental regulations and organizational policies are outside the scope of this document. 7.1.6 provides an example of policy interoperability.

5.2.1.7 Issues affecting cloud interoperability

The most significant aspect of interoperability is the mutual understanding of the semantic data and behavioural interoperability which express concepts from a domain of interest.

Full interoperability between two interacting systems requires that all interoperability facets are satisfied. However, practically speaking, two systems can still interact successfully even if interoperability is not possible to achieve for all facets. However, some interoperability facets are more challenging than others when there are mismatches between the systems involved. It is worth noting that a mismatch in one interoperability facet does not imply that other interoperability facets do not match; the facets are chosen because they are largely independent of one another.

For the transport interoperability facet, if one system communicates using a REST HTTP protocol while another system communicates using the MQTT protocol, transport interoperability might still be achieved by using a protocol adapter between the systems, such as an Enterprise Service Bus (ESB).

Similarly, if the two systems differ in relation to the syntactic interoperability facet, it may be possible to enable them to interoperate using a syntax translator; an example is a syntax mapping between data encoded in XML versus data encoded in JSON.

Challenges related to the semantics of data, the intended use and the organizational realities of people and processes and the constraints of legal or regulatory frameworks tend to be far more difficult to address. For example, transport interoperability can make it possible to deliver data from one system to another but policy, legal or regulatory restrictions may make the data practically unavailable. A lack of agreement on governance structures may impose legal risks that prevent the sharing of that data.

Systems that differ in data semantics pose significant issues for interoperability. If two systems have different types of data artefacts or the meaning of data artefacts differs between the systems, the data from one system has no meaning or is unusable by the other system. In addition, it might not be possible to create semantic adapters to enable the two systems to connect meaningfully. It might be possible to create metadata or semantic mappings to provide a form (full or partial) of semantic equivalency[2].

In order to achieve successful behavioural interoperability, it is necessary that the processes or activities of the CSC's systems are aligned with the processes or activities of the relevant cloud services, otherwise, the CSP's cloud services may not provide the features and functionalities expected by the CSC. Lack of behavioural interoperability between two systems can be a very significant barrier to enabling full interoperability between the systems. The implication is that the actual behaviour of one system does not match the expectations of the other system, even if the service interface (or API) matches between the systems. It might be possible to create some form of behavioural adapter to deal with the behavioural differences but this can be a significant challenge for more complex behavioural mismatches.

Policy interoperability can be one of the most challenging and difficult to resolve. If there is a legal prohibition on a CSC using a cloud service because the service runs in a different jurisdiction, for example, then it is not possible for a CSC to use that cloud service even if all the other facets of interoperability are satisfied. CSC policies concerning data placement, e.g. for sensitive data, can also be a significant barrier to policy interoperability (as in ISO 9241-171:2008, 3.2). CSC enterprise policies can also have an impact, for example, where specific accessibility capabilities are required. In some cases, the CSC might be able to negotiate with the CSP to offer the cloud service in a different way that overcomes the regulatory or policy issues, for example, a public cloud service might be alternatively offered as a private cloud service (with dedicated resources for that customer) in order to satisfy the security policies of the customer.

5.2.1.8 Summary of cloud interoperability facet model

Table 1 — Summary of different facets of cloud interoperability

Facets	Aim	Objects	Requirements	Examples
Transport	Data transfer between systems	Signals	Protocols of data transfer	REST-based HTTP/S, MQTT
Syntactic	Receive data in an understood format	Data	Standardized data exchange formats	JSON, XML, ASN.1
Semantic data	Receive data using an understood data model	Programmatic interface	Common interpretation of data model	OData, shared understanding and meaning, OWL
Behavioural	Obtain expected outcomes to service requests	Information	Behavioural models for the cloud service	UML models, pre and post conditions, constraint specifications
Policy	Assurance that interoperating systems follow applicable regulatory and organizational policies	Regulatory and organizational polices and interoperation context	Conditions and control for use and access	Customer security policies, restriction on cross-border data transfer, regulations controlling PII

5.2.2 Cloud data portability facet model

5.2.2.1 General

Cloud data portability is the ability to transfer data from one cloud service to another cloud service or between a CSC's system and a cloud service using a machine-readable format. Like interoperability, data portability can be observed from different facets, where each facet focuses on a single dimension. To achieve data portability, all facets need to be understood and mutually agreed upon or sufficiently understood so that it is clear what facet might need attention when porting data.

This document defines three facets of data portability within the context of cloud computing. These facets are called data policy, data syntactic and data semantic as shown in Figure 5. This model is based on the cloud interoperability model in 5.2.1, adapted to focus on the different concerns of cloud data portability.

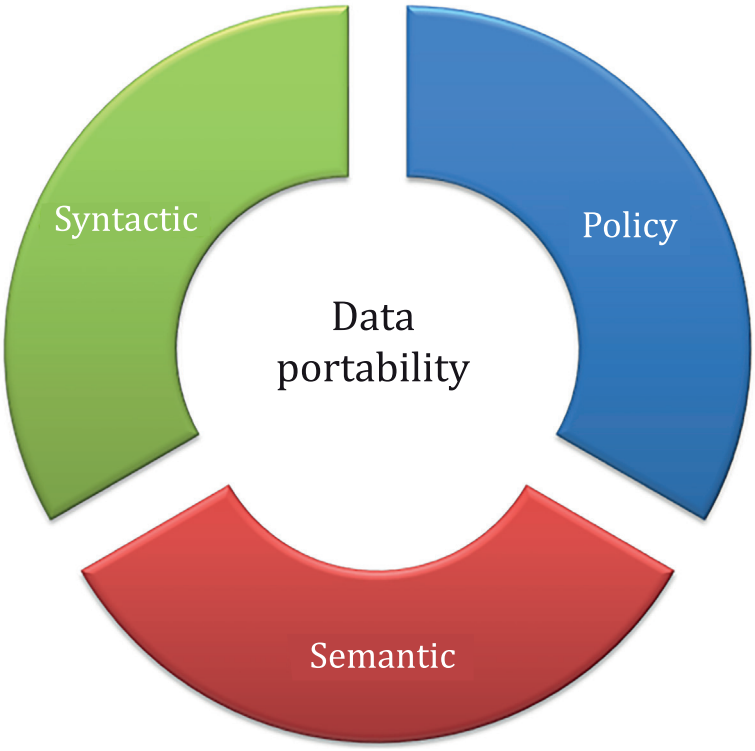


Figure 5 — Facets of cloud data portability

Unlike the cloud interoperability model, a transport facet is not included in the cloud data portability model as the process of transporting data between two systems is a separate matter which can be addressed in a number of ways.

5.2.2.2 Data syntactic portability

Data syntactic portability is defined as transferring data from a source system to a target system using data formats that can be decoded on the target system, using a particular syntax for encoding the data, such as XML or encapsulating the data in a packaging format, such as Open Virtualization Format (OVF) [3] or Zip[8].

5.2.2.3 Data semantic portability

Data semantic portability is defined as transferring data to a target such that the meaning of the data model is understood within the context of a subject area by the target. A (logical) data model, sometimes called a meta-model, expresses the data items, their attributes, logical structures and relationships between data items. Data models in a cloud context are dictated by the type of cloud service offering. At the infrastructure level, data models concern VMs, containers, storage and networking metadata. At the platform level, the data models concern applications intended to be installed. At the application level, the domain concepts and data model will be dictated by the application itself, such as HCM, CRM and ERP.

5.2.2.4 Data policy portability

Data policy portability is defined as the ability to transfer data between a source and a target while complying with the legal, organizational and policy frameworks applicable to the source and target. This includes regulations on data locality, rights to access, use and share data, and mutual responsibilities with respect to security and privacy between a CSP and a CSC. For some specific issues that can affect compliance for privacy and confidentiality during migration, see 5.3.4.

5.2.2.5 Issues affecting cloud data portability

The core of cloud data portability is a mutual understanding of the semantics as captured in a data model. Semantics can be encoded into different syntaxes, but providing the different parties have a mutual understanding of the data model, translations between syntaxes will result in minimal information loss. Similarly, policies may change but semantic data portability ensures consistency.

Data portability is a key consideration when considering lock-in to a specific cloud service. Although application portability (see 5.2.3) can affect the cost of migrating between systems, application migration is largely a cost issue and although economic lock-in is a real possibility, it can be mitigated. However, if key data is unavailable outside of a system, it may be impossible for a CSC to leave that system and they are locked into that system. Data syntactic incompatibilities can increase costs involved in moving data between systems but they are rarely a cause of lock-in for cloud services. It is necessary to focus on semantic portability of key data when considering the feasibility of data portability since loss of access to meaningful information in the data may make the target system unable to provide supporting activities important for CSC and CSN roles. In addition, an absence of data policy portability can make it impossible to access the data and key information.

EXAMPLE An enterprise CSC uses a SaaS offering for CRM and the commercial terms for use of that offering become unattractive compared with other SaaS offerings. The cloud service customer data held by the SaaS offering is crucial to the enterprise's operation. However, the CSC discovers that the data semantics are specific to their current CRM service and cannot be easily ported to a competitor.

In many such cases, porting will be very difficult. The structure of the data is often designed to fit a particular form of application processing which has developed over years of operation and a significant transformation is needed to produce data that can be handled by a different cloud service.

Data formats define syntax and convey semantics, so it is important to consider the role of data formats within data portability. In addition to interfaces, application programs and software packages acquire, store and process data using structures, which are optimized by the software developer. Following the principle that compatible interfaces are important in a cloud environment, two implementations of the same service do not have to be created in the same way and can store data very differently. Data storage methods and formats will also need to change as innovation brings new features or performance improvements to a service. It is likely that internal data storage methods and format will change dramatically over the life cycle of the service. In general, it is not possible to “freeze” the internal data storage methods or formats without also blocking future innovations.

Without proper definitions of import and export formats, a set of data from one cloud service will probably be meaningless when imported into another cloud service. This problem clearly has an impact on data portability between CSPs. Software is available in many different business domains. Therefore, data interchange formats will need to be considered for specific domains such as administration, finance, healthcare, etc. This issue is of particular concern at the SaaS level.

Regardless of the domain, special attention has to be given to personal information. For example, according to Reference [16], the data subject has the right to receive personal data in machine-readable, structured and commonly used formats. Personal data can be described by a number of categories. Information regarding these data categories can be found in ISO/IEC 19944. Furthermore, data in any category may provide or contribute to information that can be linked to an individual, referred to in this document as personally identifiable information (PII). The extent to which individuals are directly identified in the data and how easy it is to associate a set of characteristics in the data to an individual is important to individuals, CSCs and policy makers as they assess the use of that data category. Therefore, the specification of data often includes not only the type of that data, but also a description of the degree to which the data can identify an individual (ISO/IEC 19944:2017, 8.3).

Standards that define the semantics of some elements of the data can help ensure semantics portability. For example, Dublin Core[1] schema is a vocabulary that describes web resources.

5.2.2.6 Summary of cloud data portability facet model

Table 2 — Summary of different facets of cloud data portability

Facets	Aim	Objects	Requirements	Examples
Data syntactic	Receiving data in a machine readable, structured and commonly used format	Data	Common machine-readable data format	XML, CSV, JSON
Data semantic	Assured meaning of data	Data schemas and ontologies	Mutually understood ontologies and metadata	OWL, Dublin Core schema
Data policy	Adhering to all applicable regulations and organizational policies	Regulatory and organizational policy	Agreed set of applicable regulations and organizational policies	Confidentiality levels, privacy rights, cross border transfer

5.2.3 Cloud application portability facet model

5.2.3.1 General

Cloud application portability is the ability to migrate an application from one cloud service to another cloud service or between a CSC's system and a cloud service. The objective is that once ported, the application provides equivalent functionality in the target environment as it did on the source environment. To achieve application portability, all facets needs to be sufficiently understood so that it is clear what facet(s) might need attention when porting an application.

This document defines five facets of cloud application portability within the context of cloud computing. These facets as shown in Figure 6 are application syntactic, application instruction, application metadata, application behaviour and application policy. The cloud application portability facet model is inspired by the cloud interoperability facet model described in 5.2.1 and adapted to focus on the different considerations related to cloud application portability.

Unlike the cloud interoperability model, a transport facet is not included in the cloud application portability model as the process of transporting applications between two systems is a separate matter which can be addressed in a number of ways.

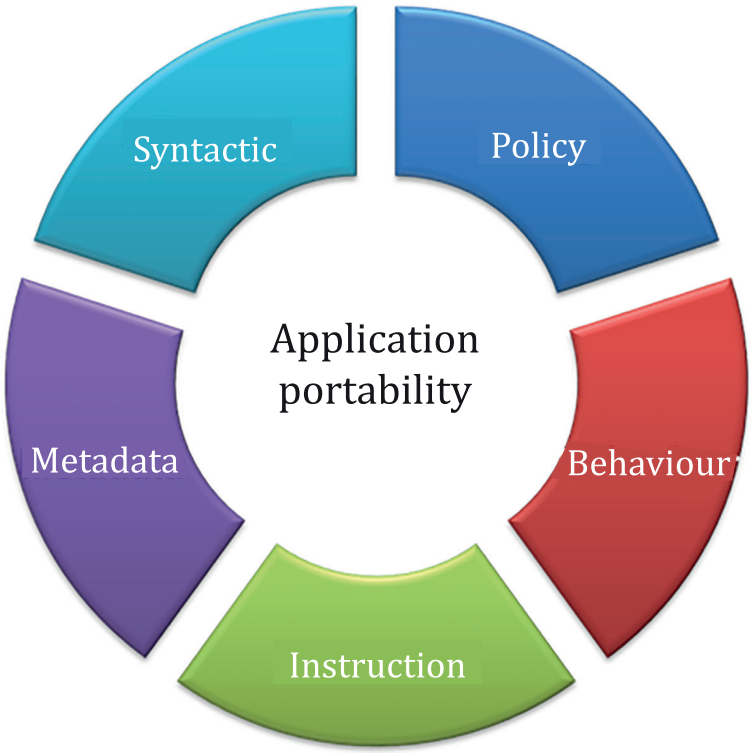


Figure 6 — Facets of cloud application portability

5.2.3.2 Application syntactic portability

Application syntactic portability is migrating an application from a source system to a target system in a format that can be decoded on the target system. Similar to data syntactic portability, application artefacts and metadata are structured according to a domain model for applications and are encoded using a particular syntax and packaging format.

5.2.3.3 Application instruction portability

Application instruction portability is migrating an application from a source system to a target system so that its instruction set executes on the target system. Once ported, the software artefacts, orchestration instructions and other scripts that comprise an application need to be executed on an infrastructure that appears to the application as similar to the system for which it was designed. This means ensuring all the necessary interpreters and execution engines are available.

5.2.3.4 Application metadata portability

Application metadata portability is migrating an application from a source system to a target system so that the application metadata is understood on the target system. Similar to semantic data portability, the domain model for an application needs to be mutually understood if an application is ported from one system to another. The domain model for an application typically includes metadata about the application, including what resources the application needs, how it might be configured, initialization data, etc.

5.2.3.5 Application behaviour portability

Application behaviour portability is migrating an application from a source to a target so that execution on the target produces equivalent results to those produced on the source. An application ported from one system to another might not exactly exhibit the same behaviour in the target system due to the differences in the execution environment. For example, fast clock speeds may cause threading and

locking issues or there may be an increase in response times to a user request causing HTTP timeouts. Such problems are usually a result of legacy application design, e.g. out-dated latency assumptions, and need to be addressed by the CSC in the application, since the CSP has no control over such behaviours. This is much less of an issue for applications developed for cloud service deployment.

5.2.3.6 Application policy portability

Application policy portability is defined as migrating an application from a source system to a target system while complying with the applicable legal, organizational and policy frameworks of both the source and target systems. Application policy portability can be affected by a number of factors, for example, lack of payment for the cloud service, lack of a license to run the application in the target system, regulations preventing running an application in a certain geography, etc.

5.2.3.7 Summary of cloud application portability facet model

Table 3 — Summary of different facets of cloud application portability

Facets	Aim	Objects	Requirements	Examples
Application syntactic	Received application in an understood format	Application	Common packaging format	Zip, tar, jar
Application instruction	Ability to execute application instructions in functional equivalent manner	Application instructions	Supported runtime environment	C++, Java, C#, BPEL,
Application metadata	Mutual understanding of environmental dependencies needed for proper execution of application	Application metadata	Shared metadata model	XML, JSON, YAML
Application behaviour	Ability to execute an application to produce expected results	Application functional and non-functional requirements	Shared assumptions on environment	Application test suites
Application policy	Agreed set of regulatory and organizational policy constraints on application use	Regulatory and organizational policy	Conditions and control for use and access	Licenses, applicable regulations, enterprise policies

5.3 Key challenges related to interoperability and portability in cloud computing

5.3.1 General

The following key challenges need to be carefully considered, even in cases where identical software is used in both the source and the target system.

5.3.2 Security

Security is a key concern for all CSCs when using cloud services, both in terms of interoperability and also, in terms of data portability and application portability.

For interoperability, concerns include the security controls, for example, as defined in ISO/IEC 27000 series of standards, applied to the interfaces between the CSC's systems and the cloud service. These controls include the confidentiality of communications between the CSC's systems and the cloud service, typically enabled using encryption of some form. Confidentiality controls can include encrypted protocols such as HTTPS or Transport Layer Security (TLS) or can take the form of a Virtual Private Network (VPN) between the CSC's systems and the cloud service. There are a number of standards in this area, which are likely to make interoperability possible from the point of view of this security concern.

A second concern for interoperability is that of identity and access management, covered in 5.3.3.

For data portability, the movement of cloud service customer data from a CSC's system to a cloud service or from one cloud service to another cloud service raises a series of security concerns. The essential question is whether the security controls in place in the target cloud service meet the CSC requirements for the data concerned. The CSC needs to classify the data; more sensitive data need a higher level of security controls applied. Factors affecting data classification include:

— whether the data contains PII;

— if the data contains PII, whether any of the PII is sensitive data;

— whether the data contains confidential data, such as financial records or data supplied by third parties under a restrictive license (e.g. data which is subject to digital rights management);

— whether the data is subject to regulation and if so, what restrictions or requirements are imposed by the regulation.

Data of all classifications is likely to need security controls in place to assure availability. How this is done can vary but suitable backup and restore facilities are typically required and/or data replication and redundancy capabilities. How these capabilities are provided can vary widely. In some cases, the capabilities are built into the cloud service and in other cases, the CSC needs to set up the capabilities.

Data classified as low risk might be placed into a cloud service with a relatively small set of security controls, although even this data is likely to need protection from tampering or destruction by unauthorized users. Data classified at a higher level of risk is likely to need more security controls. These controls are likely to include encryption of the data at rest, e.g. when stored in a database, encryption of the data when in motion over any communication link and granular control over access to the data (see 5.3.3).

Once encryption is necessary, consideration needs to be given to the form of encryption, e.g. does the encryption meet the requirements of a standard such as Federal Information Processing Standard (FIPS) 140-2[17] and also to the mechanisms used to handle the encryption keys. Encryption key handling can include key storage services and the use of hardware encryption modules. It is possible that the encryption capabilities differ between the CSC's systems and the cloud service or between cloud services and these differences shall be accounted for during the porting process.

More sensitive data usually also requires close monitoring, with requirements for recording all access to the data and recording all changes made to the data.

A key question for all these controls is who is responsible for applying and operating the controls. This is likely to vary depending on the cloud capabilities type of the cloud service. For application capabilities cloud services, most of the controls are likely to be in the hands of the CSP, although the CSC might need to select the appropriate configuration of the cloud service. For infrastructure capabilities cloud services, many of the controls are likely to be in the hands of the CSC, except for general data centre controls such as physical security. For platform capabilities cloud services, there might be a mix of responsibilities depending on the platform capabilities used by the CSC.

For application portability, key security considerations concern the security controls which enable the CSC to assure that the ported application artefacts are not subject to tampering, destruction or theft. There are also the security controls relating to the operation of the application (firewalls, authentication, encryption and so on). Protection of the application artefacts relates to both protection when in motion between CSC's systems and the cloud service and also at rest when stored within the cloud service. Encryption and strict access control are capabilities likely to be necessary. For the operation of the application, it is typical that the CSC needs to arrange for all the necessary capabilities to be in place (as applies where the application runs on CSC in-house systems). For some cloud services, it may be the case that some security capabilities are supplied by the cloud service, either automatically or through configuration, e.g. firewalls, in this case, the CSC needs to understand the capabilities available including their responsibilities for configuring and operating them.

In all cases, interoperability and portability are improved where the target cloud service meets the security requirements of the CSC, ideally with minimal configuration and change on the part of the CSC.

5.3.3 Identity and Access Management (IdAM)

Cloud services employ an identity and access management system to control access to the interfaces offered by a cloud service and also to control access to resources inside a cloud service. An IdAM system needs to know who everyone using a cloud service is (people, groups, organizations and objects). Note that identities apply not only to a human person, but are also required for other identified entities such as groups, departments, mailing/security lists, job roles, teams, projects, companies, subsidiaries, devices, computer domains, policies, etc. Exactly what the full set of possible identities cover and the format used for them can vary from one cloud service to another.

A major concern for cloud service interoperability and to portability relates to the IdAM system used in conjunction with the cloud service. It is typically the case that a CSC has an IdAM system which is used for their existing systems and which is used for their applications when running in-house. When porting the application(s) to a target cloud service, a CSC faces the choice of either switching to use an IdAM system supplied by a cloud service or else continuing to use their own IdAM system in cases where a cloud service IdAM system supports delegation of authentication requests to a CSC's IdAM system. In both of these cases, the aim is to have a single location for the control of the identities of a CSC's users. This improves security since only one IdAM system needs updating for significant events such as the removal of access for a user. The alternative, less favoured approach, is to use two separate IdAM systems, one for a CSC's system and a separate one for cloud services. This approach introduces a security risk in that operations have to be performed separately against the two IdAM systems with the possibility that they get out of synchronization.

IdAM interoperability is supported by a series of standards, including Lightweight Directory Access Protocol (LDAP)[18], OAuth[19] OpenID Connect[20] and Security Assertion Markup Language (SAML) [21] and these can help support the key requirements of federated identity management and Single Sign-On (SSO).

Where an IdAM system is changed in porting to a cloud service, a CSC needs to give careful consideration to the effort required to migrate user identities to/from a cloud service dedicated IdAM system. In migrating user identities, the identifiers used for a given entity can change

This becomes significant in cases where the digital identifiers are used by CSC applications or within CSC datasets. If the digital identifiers change as a result of a changed IdAM, this could necessitate significant and risky updates to the applications and datasets.

When considering data portability and application portability, it is necessary to remember that data files do not stand alone, even when wholly adherent to a document format. They have metadata associated with them in terms of object ownership, Access Control Lists (ACLs), audit history, change tracking, etc. Some document files will have detailed information of who made which edits, who has made and read comments, etc. All of these features depend on digital Ids as managed by the IdAM system. File metadata might or might not be portable even where the files themselves are portable.

Similar sorts of metadata might also be relevant to other types of artefact involved in portability, including database tables and records, virtual machines, network connections and other virtualized resources.

Therefore, moving to another cloud service requires that either

1) all digital identities in the original system are recreated in the target system with the same digital identity values (which also implies that both systems need to be using the exact same digital identity format), or

2) every place where an identity is used in any data, metadata, software code or application be changed so that it is replaced with the correct identity for the new system. Reliable mappings will need to be created to provide correspondence of identifiers across participating systems. This correspondence will apply to all domains and facets necessary for application and data portability.

5.3.4 Security during migration

In addition to the security considerations when a cloud service is in use, there are a number of serious operational security risks that apply during the migration to a target cloud service. For example,

— if security access is relaxed during or due to migration, perhaps to ease the migration, unauthorized systems or interfaces might have access to things they should not. Even if nobody abuses this, the CSC may have a major compliance issue;

— if security is tightened, there is a risk that authorized people/systems are unable to access the resources needed to do their job. This can impose a significant burden on the CSC's cloud service administrators, since they will have to follow normal approval process for re-granting such access, otherwise, they risk loosening the security as noted above.

The issue of the impact of modified digital identifiers mentioned in 5.3.3 applies.

Another security issue to be considered is when migrating data and metadata and updating ACLs as noted in 5.3.3. In many cases, this requires data to be decrypted, modified and re-encrypted. This implies access to the necessary public/private keys from the old and new systems, which also raises security and compliance questions. Even where not actually encrypted, some files and other data objects could be digitally signed to assure their authenticity, which could also cause issues during migration.

5.3.5 Dynamic migration

Traditional host migration is often achieved by shutting down the current service, taking a snapshot of everything, uploading it to the new service and going live on the new service, often over a weekend or other natural break. The scale of migration to and from globally deployed cloud services frequently preclude such an approach. Additionally, the sheer volume of data that needs to be moved can preclude a timely "snapshot and restore" approach.

This means that the migration has to be much more dynamic and gradual, with users and assets being moved from the source system to the target cloud service in such a way that individual users and assets do not suffer any appreciable loss of access. This also means that users who have not yet migrated still need access to objects that have already migrated and users who have been migrated can still access objects that have not yet migrated. Delivering reliable and consistent service during a migration can be challenging and expensive, but does avoid a "big bang" approach which has the potential to impact the entire organization if any issues arise.

5.3.6 Interfaces, APIs and interoperability

Interoperability applies to three types of interfaces or APIs associated with each cloud service, the service interfaces, the administration interface and the business interface, as discussed in 6.1.

For cloud services as a whole, there are relatively few standards that apply to these three interfaces, which can make interoperability a challenge. For the transport and syntax facets of interoperability, it has become common for cloud service interfaces to be based on TCP/IP, HTTP protocols and either JSON or XML formats using REST principles, which enables interoperability of these facets to be relatively straightforward.

Interoperability of service interfaces is a very big challenge for application capabilities type cloud services. For application capabilities, there is relatively little agreement on the behavioural and semantic facets of the service API. This can make migration to the target cloud service a major problem, even where migration takes place from another cloud service which deals with equivalent capabilities. Considerable mapping might be required in the CSC systems which interact with the cloud service in order to migrate.

Interoperability of service interfaces for platform capabilities type cloud services also involves relatively little agreement relating to the behavioural and semantic facets of the service interface. There are some cases where different cloud service providers utilize the same underlying software for their

cloud services and in these cases, the service APIs can be the same between source and target cloud services, although this is a small part of the market. Container technologies such as Open Container Initiative[1)] can assist in simplifying the migration of applications packaged in their format, even where the details of the service API used to deploy and control them differ.

Interoperability of service interfaces for infrastructure capabilities type cloud services is typically somewhat better, in two respects. The behavioural and semantic aspects of this type of cloud service are generally relatively common between different cloud services, this makes it relatively straightforward to map between the different APIs for different cloud services. Similar considerations apply to VM images, where there is widespread support for a number of commonly-used VM image formats.

For the administration and business interfaces, there is relatively little commonality at the behavioural and semantic facets of interoperability, meaning that interoperability is generally limited for these interfaces.

5.3.7 Open source

A case is sometimes made for preferring open source technologies, especially for infrastructure capabilities types of services, e.g. IaaS, on the basis that this will ease interoperability and portability where identical interfaces and run time environments can facilitate interoperability and portability.

It should be remembered, however, that there are no “silver bullets” for solving all of the issues discussed in this document. As noted in 5.3.6, many common interfaces and formats are becoming widely adopted in both open source and proprietary cloud service implementations and interworking between popular interfaces and formats is a business necessity for customers. The choice of an open source or proprietary technology should therefore be made on the basis of business justification, including careful examination of all the issues described in this document.

6 Interoperability and portability considerations related to cloud capabilities types

6.1 General

To understand the technical feasibility and cost of interoperability and portability in relation to cloud services, it is useful to group together scenarios and use cases so that they can be considered together with sufficient detail. This clause provides a framework for doing this using cloud capabilities types defined in ISO/IEC 17788 and utilizing the architectural concepts described in ISO/IEC 17789. This clause describes how the general space of interoperability and portability of cloud services can be subdivided into categories with common properties.

It is useful to start with a set of functional component diagrams which are based on ISO/IEC 17789:2014, Figure 9-2, modified to add elements necessary to understand cloud interoperability, cloud data portability and cloud application portability. The functional components are described in ISO/IEC 17789:2014, 9.2 except for the additional components introduced in this clause. These are Figure 7, Figure 8 and Figure 9.

1) https://www.opencontainers.org/

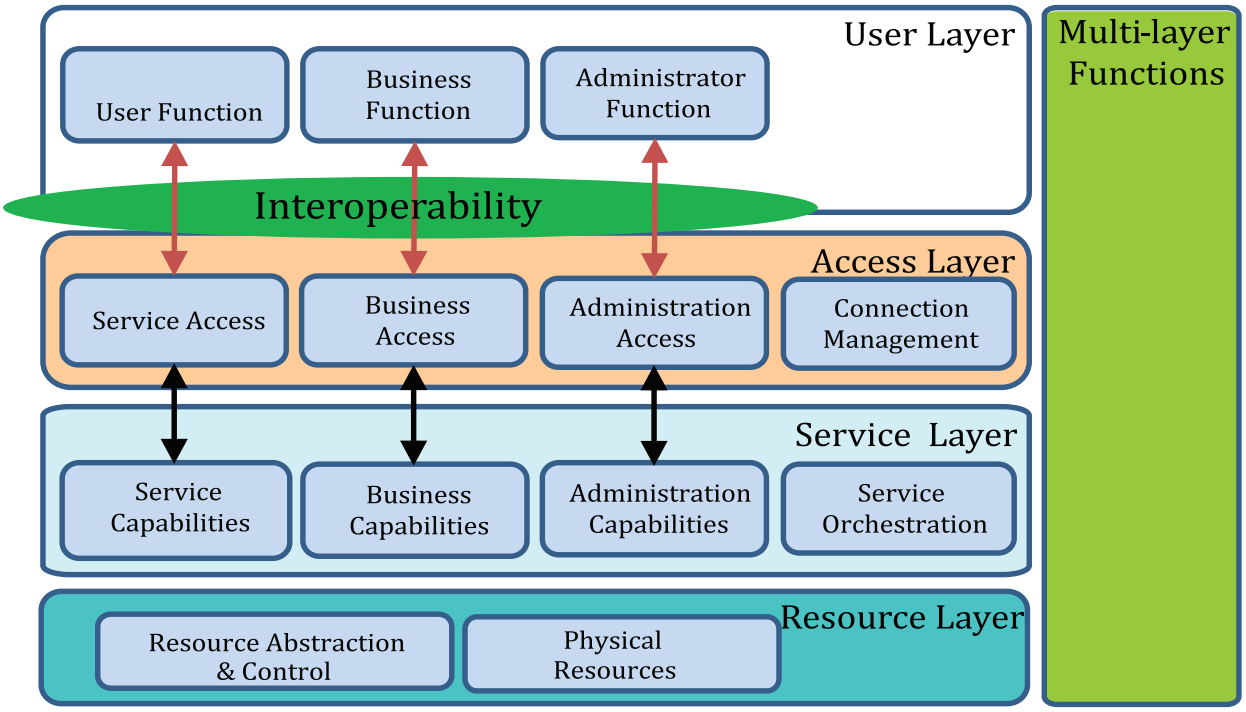


Figure 7 — Components of cloud computing interoperability

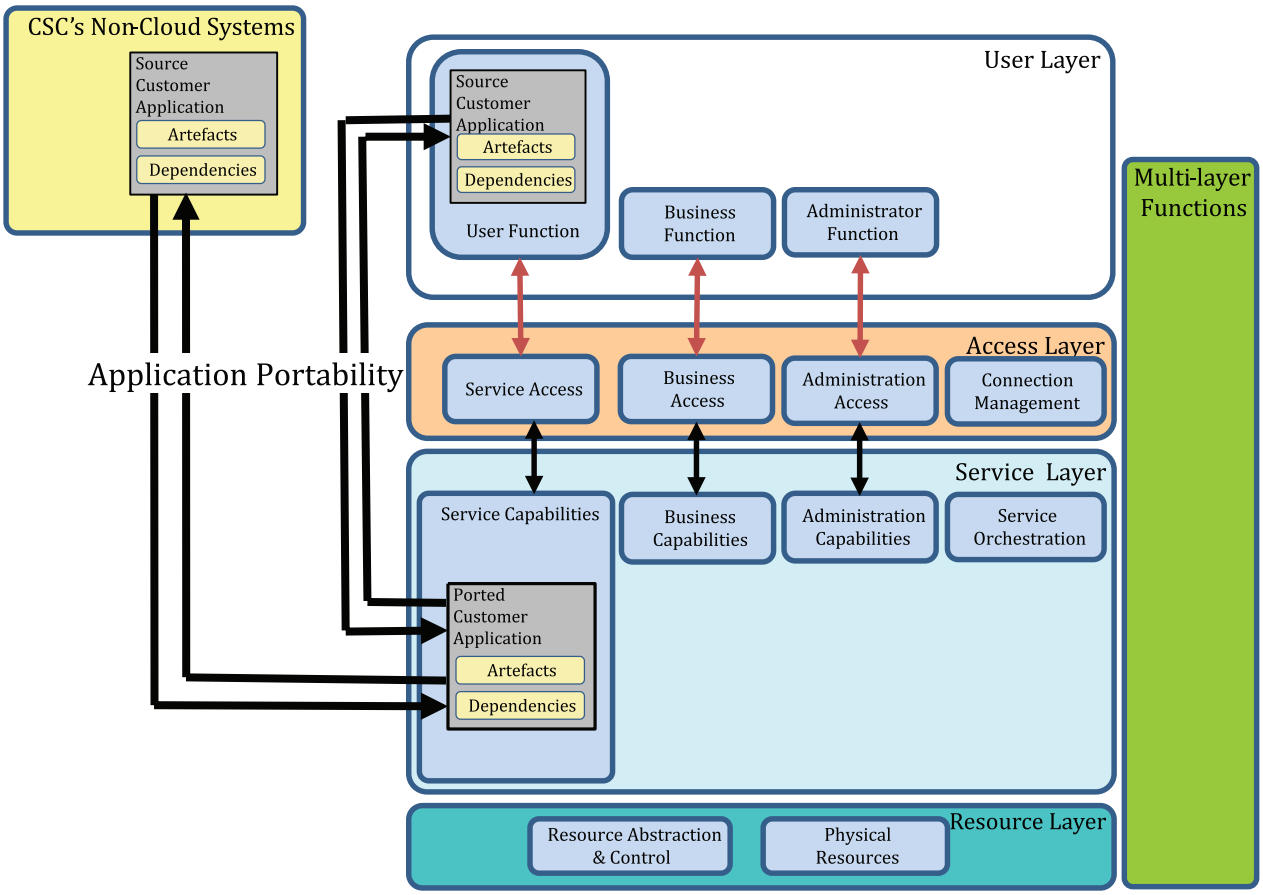


Figure 8 — Components of cloud application portability

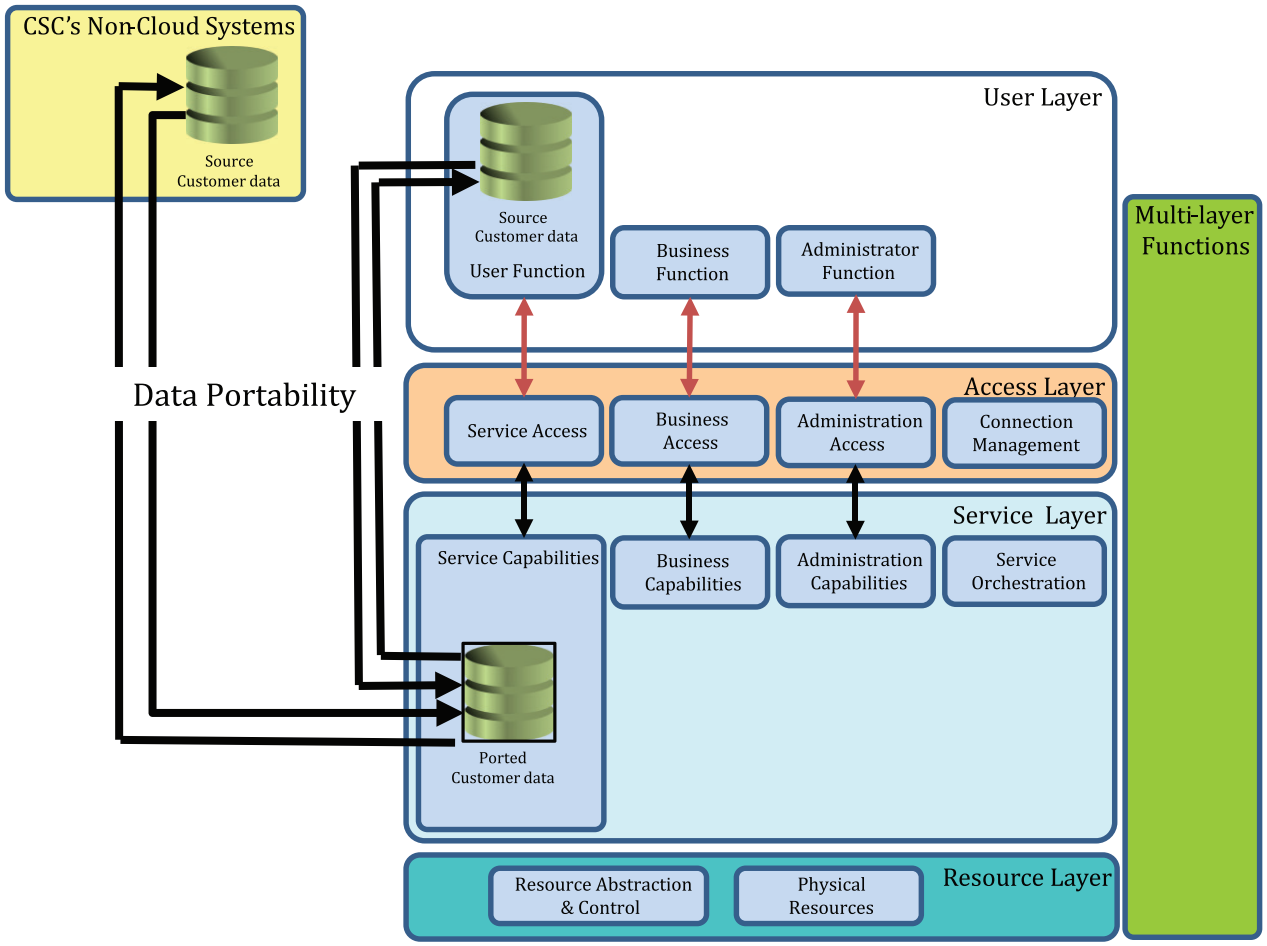


Figure 9 — Components of cloud data portability

Figure 7, Figure 8 and Figure 9 are simplified in a number of ways to aid understanding.

— The multi-layer functions are reduced to a single high-level component at the right-hand side.

— The resource layer and access layer are flattened to provide more space for the other layers.

Figure 7, Figure 8 and Figure 9 are extended to add components useful to understanding cloud interoperability, cloud data portability and cloud application portability.

a) A box labelled "CSC's Non-Cloud Systems" is added at the top of Figure 8 and Figure 9. This represents the CSC's systems which contain the CSC's applications and datasets.

b) In Figure 8, a customer application is shown in the user function component of the user layer. As mentioned in ISO/IEC 17789:2014, 9.2.1.1, "In some cases, the user function functional component could be as simple as a browser running on a user device. However, in other cases, it might involve a sophisticated enterprise system running business processes, applications, middleware and associated infrastructure". So, user function can contain an application which the CSC may wish to port to a cloud service.

c) In Figure 8, a customer application is in the CSC's Non-Cloud Systems box, this represents an application originally running in a customer system, not associated with any cloud computing, which the CSC may wish to port to a cloud service.

d) In Figure 9, a customer data component is added both to the CSC's Non-Cloud Systems box and also to the user function component, this represents one or more datasets that the CSC may wish to port to a cloud service.

e) In Figure 8 and Figure 9, the service capabilities component in the service layer is expanded to provide space for two added components within it:

   1) a customer application component, which represents a customer application ported to run within a cloud service;

   2) a customer data component, which represents one or more customer datasets ported to reside within a cloud service;

   3) it is not intended to imply that every cloud service contains a customer application and customer data; it is intended to show that this is where these components exist in the architecture when they are present.

In Figure 7, there is an oval shape labelled "interoperability" underneath the three arrows connecting components in the user layer to components in the access layer. The three arrows represent the use of the three separate interfaces offered by each cloud service: the service interface, the administration interface and the business interface. It is these three interfaces that are involved in the interoperability of the cloud service.

In Figure 8, the dashed lines show the migration of customer applications between the CSC's systems and the cloud service, which connect the components involved. Note that customer applications are shown as containing both artefacts and dependencies. These become important when describing application portability in greater detail in later clauses.

In Figure 9, the dashed lines show the porting of customer data between the CSC's systems and the cloud service, which connect the components involved.

Note that Figure 7, Figure 8 and Figure 9 represent the case of a CSC and one cloud service and interoperability and portability between the CSC's systems and that cloud service. Figure 10 represents the other major case relating to interoperability, the case where there are two (or more) cloud services. In this case, portability of both data and applications relates to moving from a source cloud service to a target cloud service and interoperability relates to the associated need for the CSC systems to work with both cloud services.

Figure 10 shows a single CSC interacting with two cloud services, cloud service 1 on the left and cloud service 2 on the right. In Figure 10, cloud service 1 is regarded as the source cloud service and cloud service 2 as the target cloud service for porting of applications and data. Note that Figure 10 has further simplification of the access layer and service layer for the two cloud services, in order to avoid cluttering the figure with components not relevant to interoperability and portability.

The porting of a CSC application from a source cloud service to a target cloud service takes place between the service capabilities components of the two cloud services, as indicated by the arrow labelled "application portability" in Figure 10. Similarly, the porting of cloud service customer data from the source cloud service to a target cloud service takes place between the service capabilities components of the two cloud services as indicated by the arrow labelled "data portability" in Figure 10.

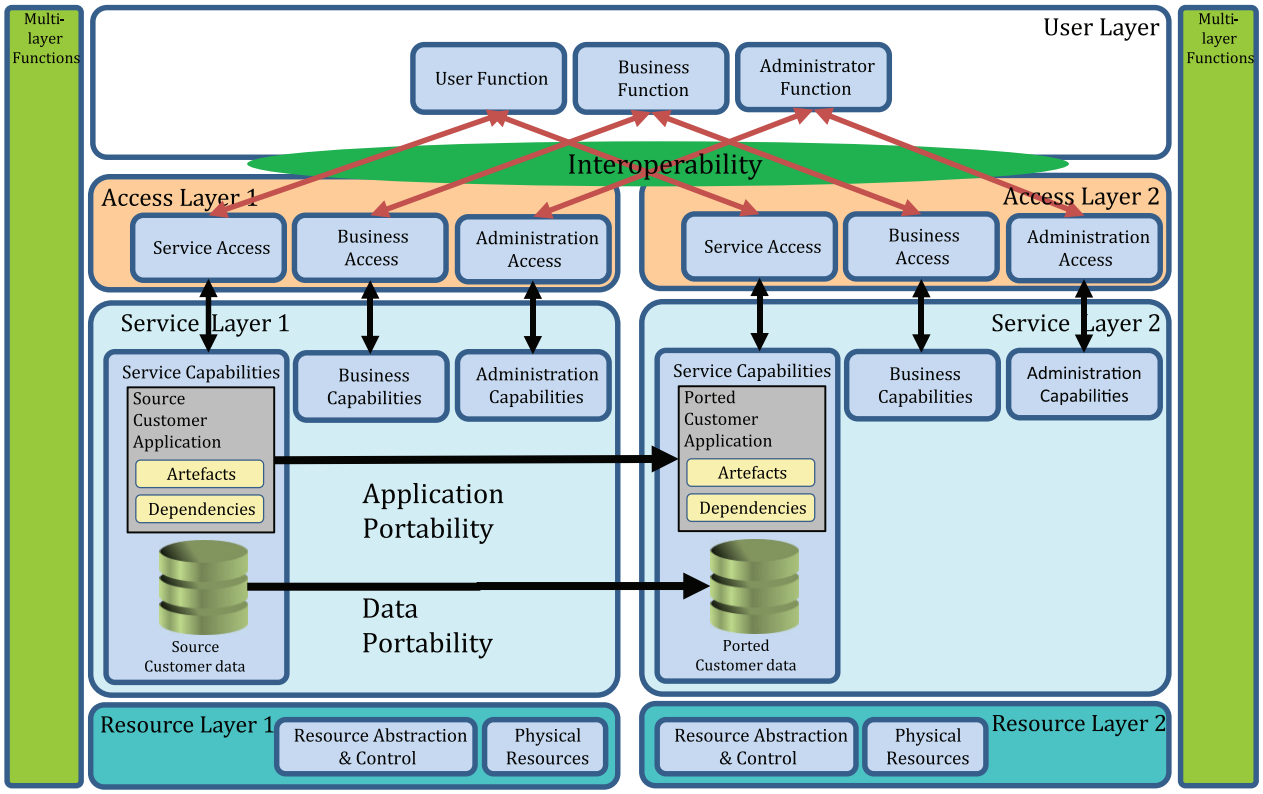


Figure 10 — Examples of relationships and interactions between activities and functional components

6.2 Functional components of interoperability

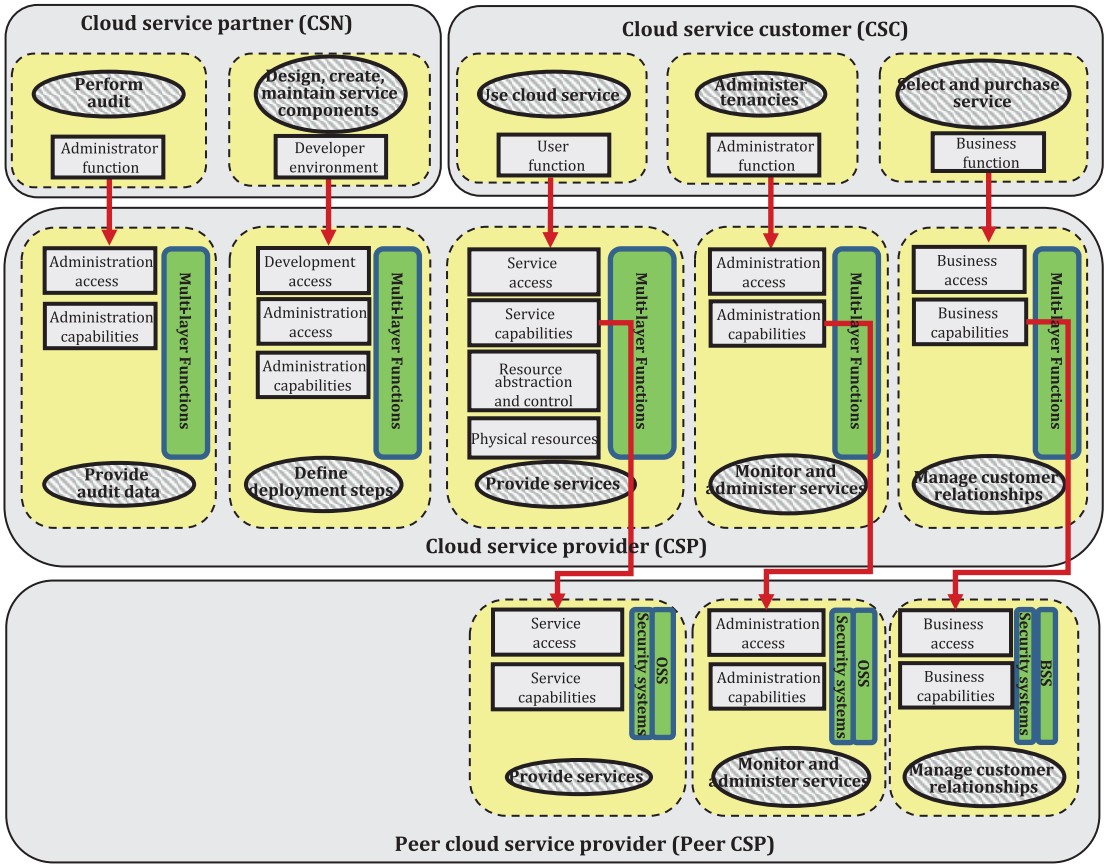


Figure 11 — Examples of relationships and interactions between activities and functional components

Figure 11 (adapted and simplified from ISO/IEC 17789:2014, Figure 10-2) shows examples of roles (CSN, CSC and CSP) using various interfaces (including administrator and business functions, developer environments and administration access) to carry out their activities. In this example, the activities include providing an audit, creating service components and managing customer relationships.

Cloud interoperability primarily relates to the interfaces presented by the service access, administration access and business access components and the use made of them by cloud service customer components. These interfaces determine the transport facet of the cloud service and also the syntactic facet with respect to interoperability. However, the implementations underlying these interfaces, the service capabilities, administration capabilities and business capabilities components, determine the semantic facet and the behavioural facet with respect to interoperability. Certain aspects of the policy facet might be determined by components in the multi-layer functions, although other aspects (especially regulatory aspects) are probably not determined by any components represented in the functional architecture.

It is important to understand that interoperability applies separately to the three main interfaces related to a cloud service. In other words, interoperability of the service interface does not imply interoperability of the administration interface.

Figure 11 also shows the components involved in one form of interoperability between two different cloud services. This is the form where the cloud service of the primary cloud service provider depends on a cloud service provided by a peer cloud service provider. In this case, it is the components of the

primary cloud service provider that interact with components of the peer cloud service provider, as shown by the arrows connecting the service capabilities of the CSP to the service access of the peer CSP in Figure 11. Similar connections apply to the administration capabilities and business capabilities of the CSPs.

6.3 Functional components of data portability

The functional components associated with cloud service customer data and which relate to its portability are shown in Figure 11.

In the user layer, customer data is held in one or more CSC's systems, which are organized as determined by the cloud service customer. These can include file systems on storage devices, object stores and databases.

In the cloud service, the customer data is present within the service capabilities component of the service layer. The storage facilities available there are determined by the cloud service itself and can include file systems, object stores and databases. These storage facilities might or might not match the storage facilities used within the CSC's systems. Where the storage facilities do match, data portability is likely to be easier, where the storage facilities do not match, the cloud service customer is likely to have to invest effort in converting the data artefacts during the porting process.

To support data portability, it is necessary for the cloud service to provide some means for the CSC to transfer customer data between CSC's systems and the cloud service. In effect, these means are the "transport" supporting data portability. This can vary widely, depending on the cloud service concerned and can include:

a) an explicit API which is part of the cloud service interface;

b) an API which is part of the administration interface of the cloud service;

c) support by the cloud service of a standard data transfer API such as FTP or SFTP;

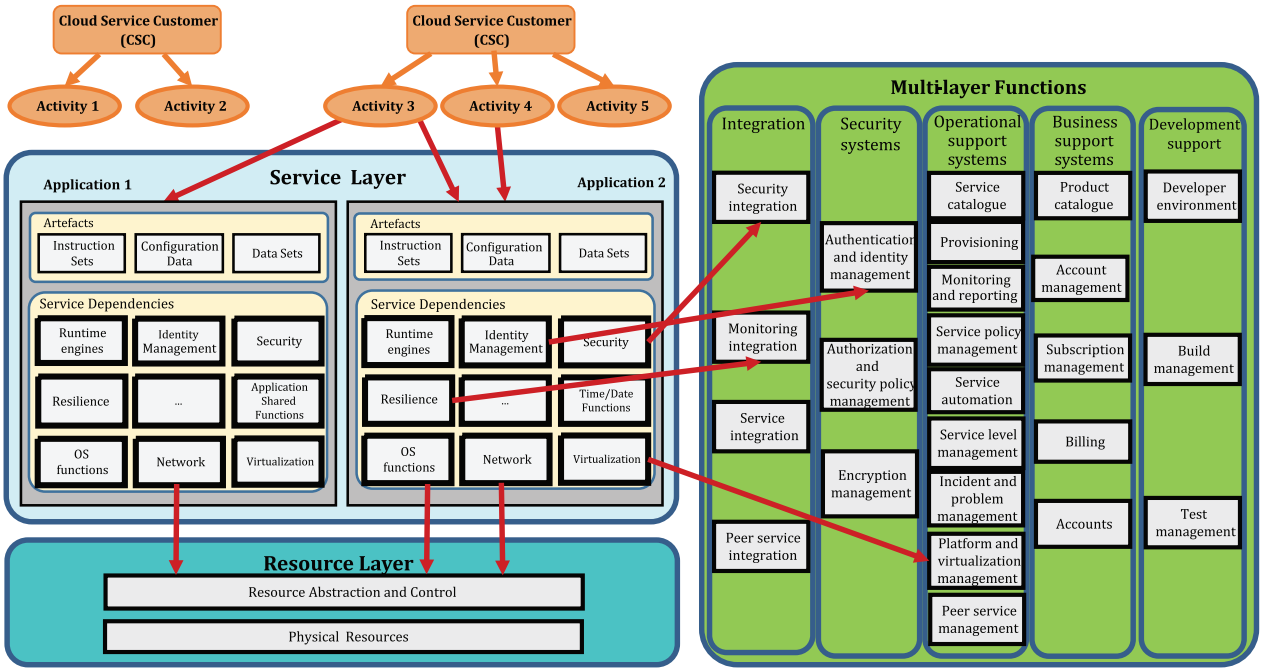
    1) more common for infrastructure capabilities cloud services.

For cloud services involving a very large volume of customer data, transfer might be offered through the transfer of data on physical media.

6.4 Functional components of application portability

6.4.1 General

An IT system typically includes data, application, platform and infrastructure elements. These can be implemented using cloud services that have cloud capabilities types of infrastructure, platform or application and can be on-premises or off-premises.



NOTE Derived from ISO/IEC 17789:2014, Figure 9-2.

Figure 12 — Service layer, resource layer and multi-layer functional components of the cloud computing reference architecture

Figure 12 shows a combination of user view and functional view and omits the user layer and access layer for clarity. It shows how the role of the CSC fulfils their business requirements by engaging in a number of activities. These activities are implemented or realized by the functional components.

Figure 12 also shows an example of two different CSCs and a number of activities. In this example, "Activity 3" is fulfilled by two applications. In reality, a particular activity may be able to be fulfilled by a single cloud service or may require any number of cloud services or applications. As explained below, these applications are composed of artefacts and service dependencies. For illustrative purposes, Figure 12 shows an example of a small number of service dependencies and how they might be resolved.

Figure 12 also shows the service layer, resource layer and multi-layer functional components of the cloud computing reference architecture.

The service layer provides the implementation of the services provided by the CSP. The resource layer provides the underlying resources for the service layer and can include host operating systems, virtual machines, virtual data storage and physical resources. The multi-layer functional components provide management, operational, integration and other supporting capabilities.

Figure 12 is derived from ISO/IEC 17789:2014, Figure 9-2 and shows some of the artefacts, service dependencies and the resources of two applications. The service dependencies of the application are shown with dashed outlines to imply that these are requirements only and choices shall be made as to how they are implemented in various scenarios. This notation is an extension of the notation presented in ISO/IEC 17789. In the examples shown in Figure 12, the actual instantiations of the functional components that provide these requirements are shown with solid outlines.

The service dependencies of an application are implemented:

— within the application itself;

— by other supporting applications;

— within the service layer itself;

— by the resource layer;

— a combination of all of the above.

A CSC (in either a business or user role) engages in one or more activities, some of which might be supported by one or more applications. As described further below, for application portability, the scope of an "application" should be clearly defined such that the necessary activities of the CSC or CSN roles are successfully supported when the application is migrated to the target service.

An application consists of a number of functional components comprised of artefacts and service dependencies of the application.

The application artefacts include:

— **Instruction sets**: This is the logic that defines the execution of the application and may include computer programming source or object code such as C++, Perl, Java or Python or instruction set definitions such as BPEL;

— **Data sets**: The application may itself use data such as a list of days of the week, countries in the world or other information which it uses in its execution;

— **Configuration, topological and state data:** Configuration data are typically targeted at a specific artefact. For example, a database artefact may have a configuration file that stipulates the maximum number of threads or users. The application may also have topological data that describes the attributes, relationships and requirements of the application, its service layer artefacts dependencies on resource layer artefacts and its requirements on multi-layer functional components, providing a holistic perspective. For example, an application may have service dependencies for specific network characteristics or artefacts may require certain types of containers to be deployed in a specific order and may optionally represent their state.

Applications have many dependencies that are generally not part of the application itself and these may include:

— **Runtime engines**: These engines interpret the original or intermediate code or run the compiled code. There might be different runtime engines for different types of instructions sets. Thus, if an application consists of some Java code as well as BPEL instructions, there are likely to be multiple runtime engines for this application;

— **Security**: Applications typically use some security services that could be contained completely within the application or could be services that are separate to the application and provided by some other system. These can include authentication services, as well as authorization services, that provide access and the authority to run parts of the application;

— **Resilience**: In order to increase the resilience of an application, it may leverage services such as fail-over or duplication so that it can still be accessed during system or other events. It is likely that these types of services will be provided by the underlying software platform;

— **Operating system services**: Applications leverage underlying operating systems services such as compute, storage and networking as well as a myriad of others.

When an application is migrated from one environment (such as an in-house data centre) to a cloud service, the scope of the application and all its artefacts and service dependencies need to be considered in that move. Most of these considerations involve comparing the existing resources with the resources that are available in the target environment.

For example, when the instruction set of one application is moved to the target environment, the runtimes in that environment need to provide the same functionality as in the source environment if those instruction sets are to be processed identically. If not, some change to the instruction set or the runtime is required.

Many of these services can depend on other services. For example, the runtime engine might require specific network capabilities or operating system services. Again, some adaptations of these services might be required to achieve comparable results.

When considering the availability and functionality of the differing services in the different environments, there will be some decisions to be made. Some services can be moved with the application itself. For example, rather than relying on external database functionality, the application and some of its dependencies can include this functionality and that could be moved with the rest of the application. This list of "included-in-the-move" services can be very extensive in the case of infrastructure cloud capabilities types or minimal in the case of platform cloud capabilities types.

In many cases, the services offered by the CSP might provide more functionality, e.g. better monitoring, resilience or improved scale and performance, or a more comprehensive approach, e.g. cross application access management, shared security or virtualization capabilities. In these cases, the original services used by the application could be replaced by the improved services offered in the new environment. Again, the use of these new services might require changes to the application artefacts or other application services.

In summary, maintaining or improving the activities of a CSC or CSN while migrating its underlying applications can require changes to many functional components of those applications and consideration of the implementation of the services that the applications require to operate. The costs, risks and benefits of those changes need to be evaluated.

6.4.2 Functional views based on capabilities types

Like Figure 12, the following functional views in Figure 13, Figure 14, Figure 15 and Figure 16 highlight various application implementations from a non-cloud environment as well as cloud implementations based on different capabilities types. The service dependencies for Application 1 in these implementations are an indicative only subset of dependencies as it is likely that the actual list for any application is much larger. Supporting applications or other functional components deployed as illustrated below in Figure 13 can satisfy these dependencies.

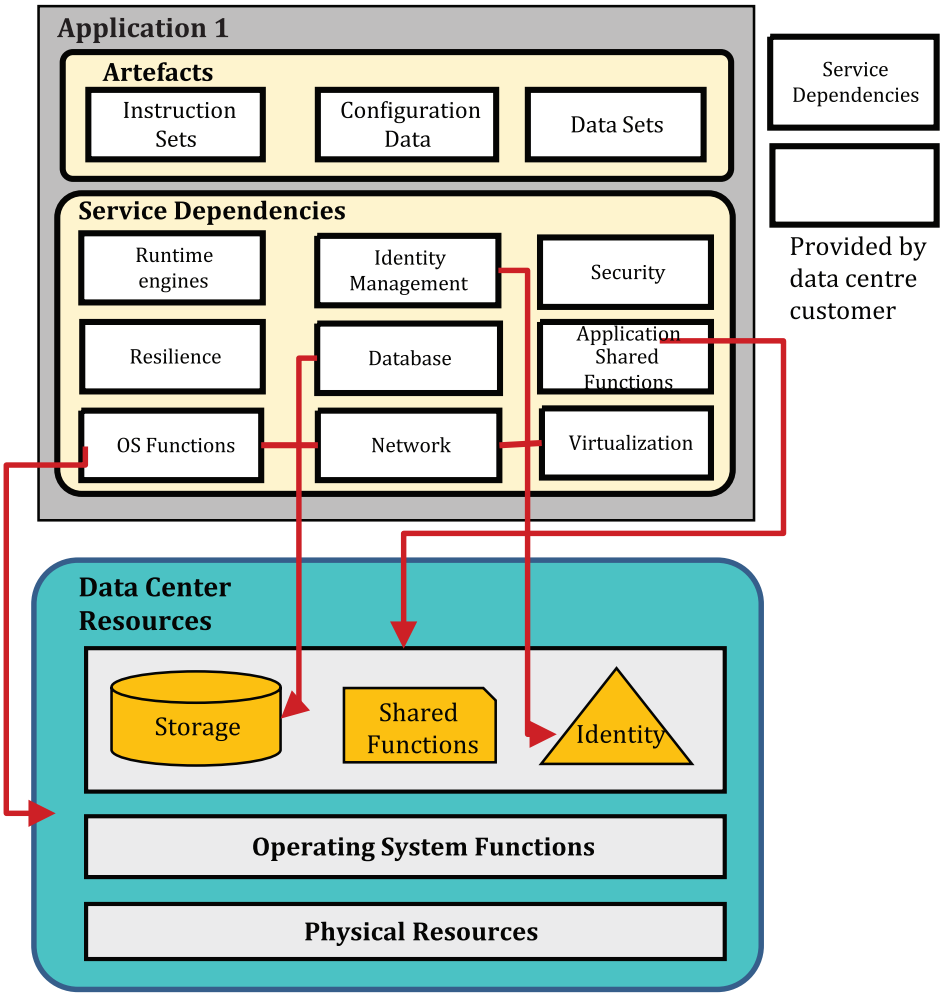


Figure 13 — Non-cloud application

In the non-cloud application functional view shown in Figure 13, all service dependencies of the application are implemented and controlled on-premises by the CSC. Some of these, e.g. security and resilience, are part of the application itself. Other requirements, including database storage, shared application functions and identity services, are implemented using the shared applications or services of the CSC's data centre.

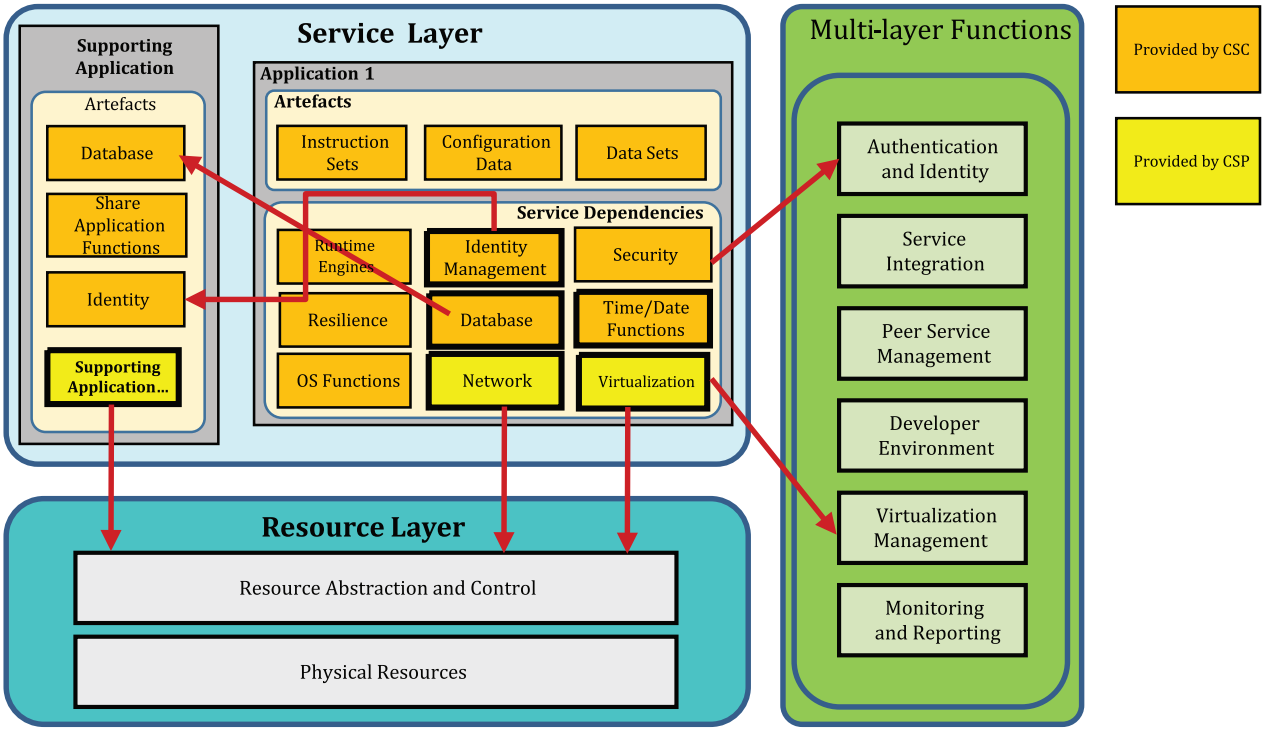


Figure 14 — Application running in infrastructure cloud capabilities type cloud service

Figure 14 shows a functional view of an application running in an infrastructure cloud capabilities type cloud service. Some of the service dependencies of the application, e.g. network and virtualization, are provided by the CSP. But most of the services need to be included in the application as in Figure 14, e.g. resilience, security. Other services, such as database storage, shared application functions and identity services are provided in other virtual machines that provide this application (and other CSC applications) with its required services (shown as "supporting application" in Figure 14). The supporting application components are ported to the cloud service environment alongside the applications which they support.

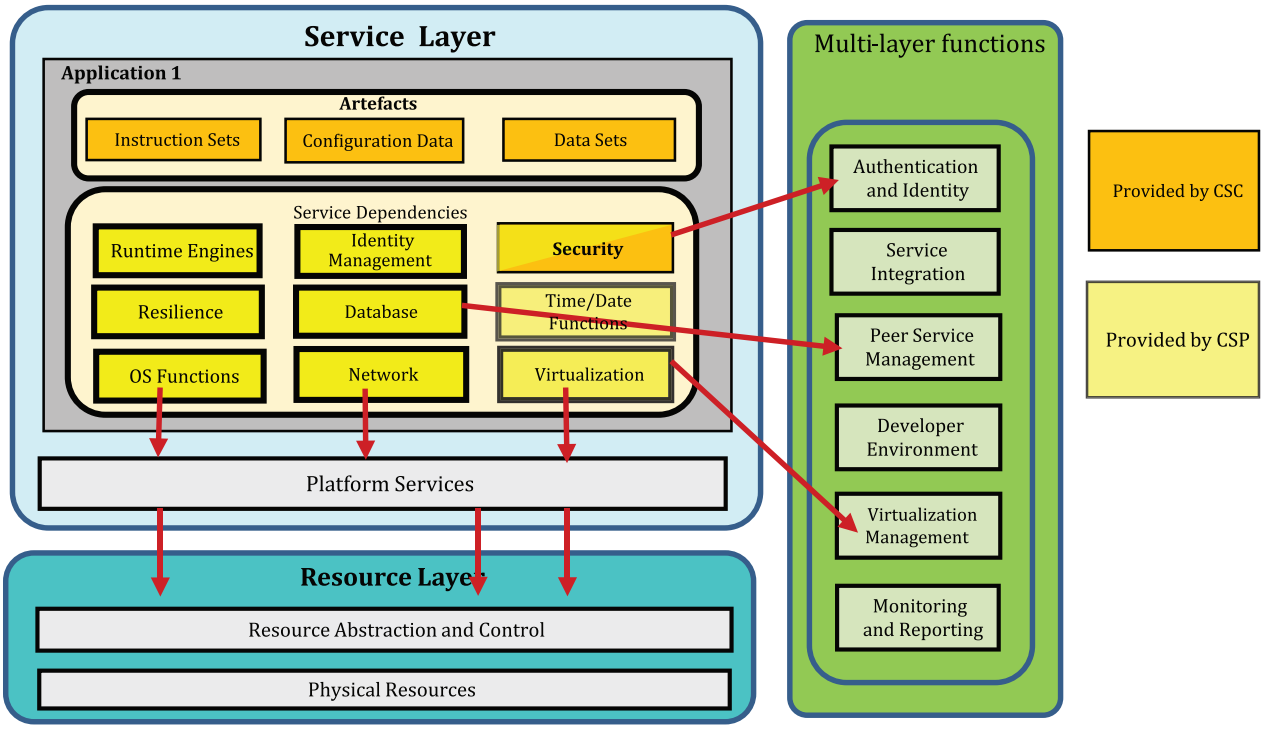


Figure 15 — Example application running in platform cloud capabilities type cloud service

Figure 15 shows a functional view of an example application running in a platform cloud capabilities type cloud service. The CSC is responsible for configuring and implementing the artefacts of the application. The actual implementation can vary depending upon CSC requirements, for example, as shown in Figure 17. However, most of the service dependencies of the application are provided by the cloud service, including identity, storage, application runtime engines and application resilience functions. These services are provided within the platform services functional component as shown in Figure 15. Some services required by the application are implemented in a mixed mode. For example, as shown in Figure 12, security services management can be implemented as a combination of application instruction sets, application data sets as well as services provided by the cloud service.

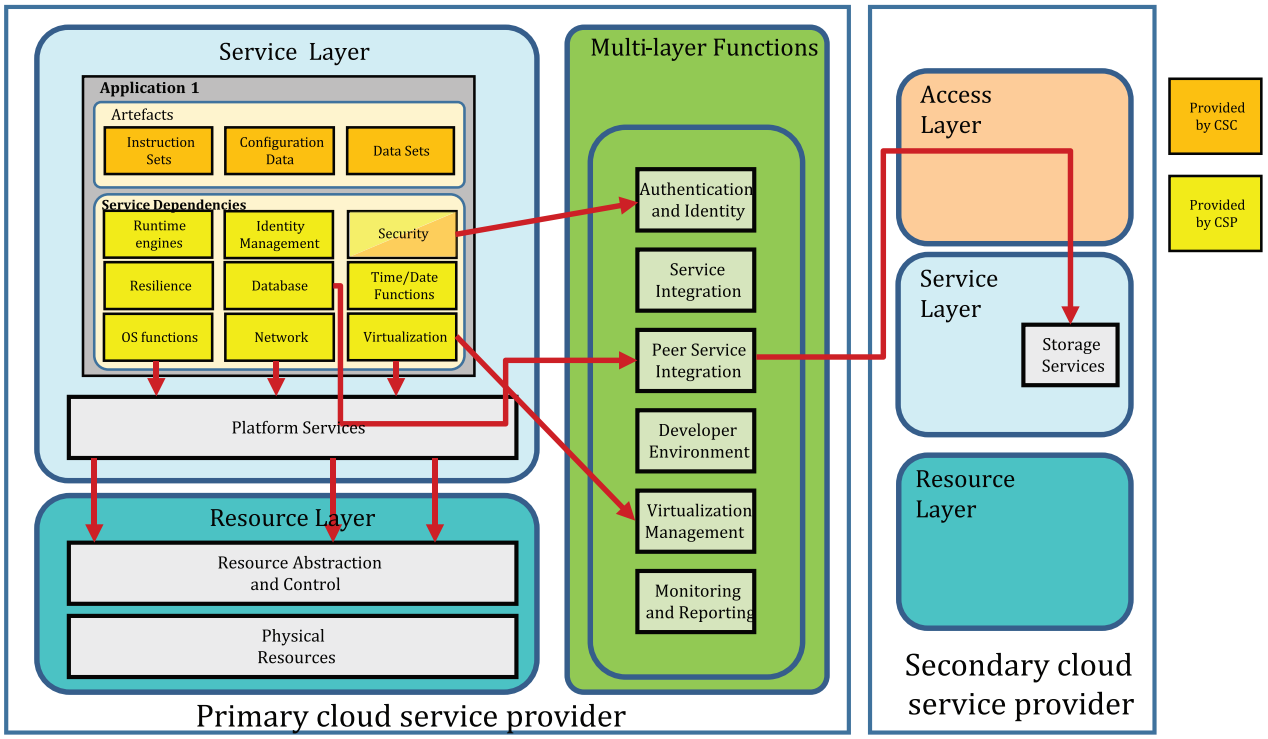


Figure 16 — Example application running in platform cloud capabilities type cloud service with primary and secondary CSPs

Extending the example shown in Figure 15 and Figure 16 shows that some platform services could be implemented by a secondary CSP. In this example, the database requirements of the application are fulfilled by a call to the primary CSP's platform services. However, this call is redirected through the peer service integration component of the primary CSP to the secondary CSP cloud service where it is actually implemented. It is also possible that the choice of the services used lies in the hands of the CSC, through explicit service resolution in the deployed application, unlike the situation depicted in Figure 16, where the choice is implicit and made by the CSP.

The particular secondary CSP that is used for this implementation could be decided in a number of ways including at design-time of the application or a run-time choice based on availability, cost, service capabilities or other variables. Alternatively, it could be decided by the primary CSP, which could be transparent to the CSC's application.

Although not shown in the examples, this redirection could also be achieved through the role of the CSP:inter-cloud provider whose activities include arbitrage, intermediation, aggregation and federation of peer cloud services.

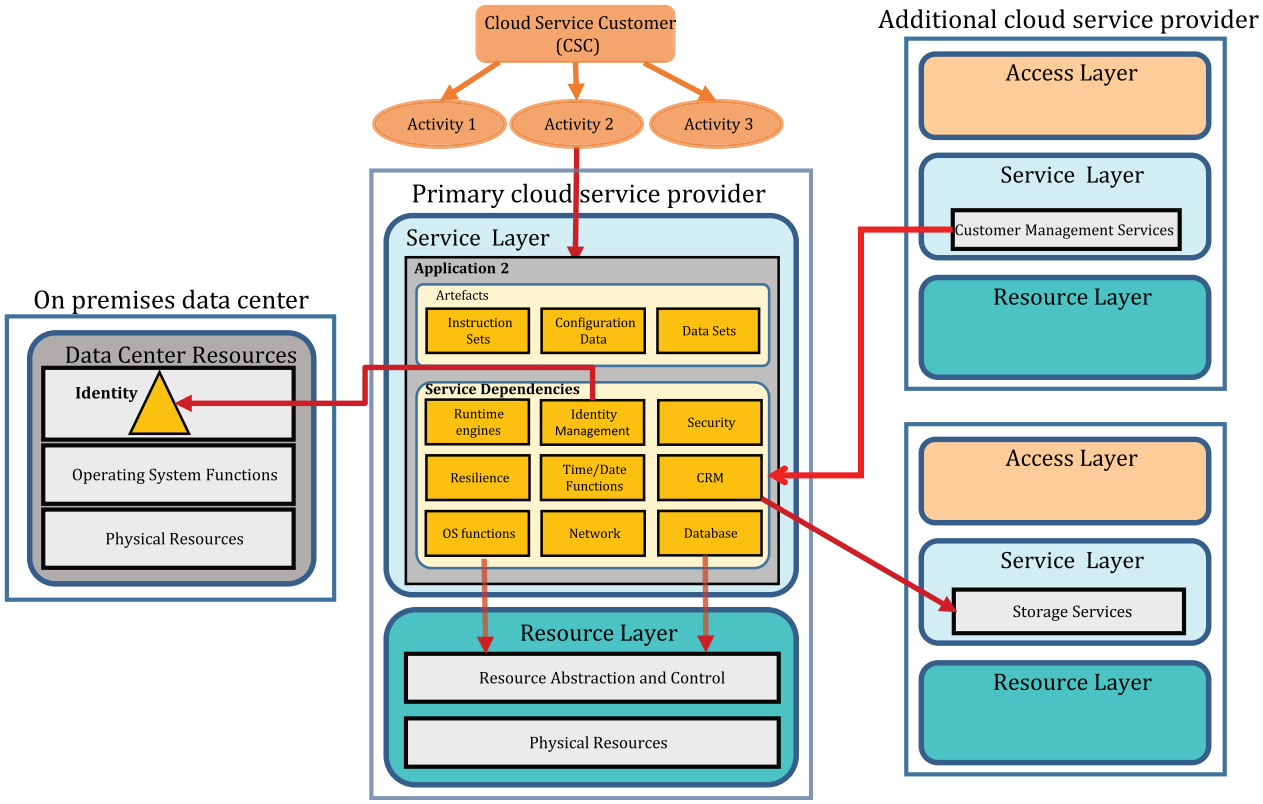


Figure 17 — Application using multiple cloud services and non-cloud data centre

Figure 17 shows an example of an application where the application service dependencies are implemented by the primary CSP, two additional CSPs (providing customer management and storage services) and by a non-cloud data centre (which is providing identity services). Figure 17 omits much of the implementation detail and is simplified to show how an application might appear to the CSC as running in one cloud provider, but its service dependencies could be implemented across a hybrid cloud as well as an on-premises data centre.

This shows that the resulting application may use a wide variety of services from many providers.

7 Cloud interoperability

7.1 Cloud interoperability types

7.1.1 General

Table 4 describes the types of cloud interoperability based on cloud capabilities types and the types of interfaces defined in ISO/IEC 17789. Each cell at the intersection of the rows and columns of the table references a subclause that describes considerations for that type of cloud interoperability.

Table 4 — Cloud interoperability types

Cloud interoperability facets	Types of interfaces involved per ISO/IEC 17789								
	Infrastructure			Platform			Application		
	Service	Business	Administration	Service	Business	Administration	Service	Business	Administration
Transport (5.2.1.2)	7.1.2								
Syntactic (5.2.1.3)	7.1.3								
Semantic data (5.2.1.4)	7.1.4								
Behavioural (5.2.1.5)	7.1.5.2			7.1.5.3			7.1.5.4		
Policy (5.2.1.6)	7.1.6								

Each cell in Table 4 is considered an interoperability type and is enumerated in the following subclauses. Some interoperability types might have similar characteristics and thus, the associated cells reference a common subclause.

The reference architecture defined in ISO/IEC 17789 includes interoperability as a cross-cutting aspect that applies to multiple elements. In the overall context of ISO/IEC 17789, interoperability occurs between functional components of a CSC or CSN and a CSP or between functional components of different CSPs.

The functional components involved in cloud service interoperability are shown in Figure 7, Figure 8, Figure 9 and Figure 10. Interoperability takes place across three interfaces between CSC components and CSP components.

— User function component interacts with the service access component using the service interface(s) offered by the service access component. The user function component enables the cloud service user to access the functionality of the cloud service.

— Administrator function component interacts with the administration access component using the administration interface(s) offered by the administration access component. The administrator component enables the cloud service administrator to perform activities including manage user identity and access, monitor activity and usage and manage faults for cloud services.

— Business function component interacts with the business access component using the business interface(s) offered by the business access component. The CSC:business manager performs activities including selecting, purchasing and managing the accounts for cloud services using the business function component.

As indicated in 6.2, interoperability considerations apply separately to each of these three interfaces. For most cloud services, the three interfaces almost certainly use different APIs, possibly involving different transport protocols and different syntaxes and data semantics. The key concept is that to achieve full interoperability, all the activities of the CSC subroles supported by the user function component, by the administrator function component and by the business function component need to be supported by the interface that each component uses to interact with the cloud service described in ISO/IEC 17789:2014, 9.2.2. They are:

— service access, which provides interface(s) that enable the CSC:cloud service user to access and use the functionality of the cloud service;

— business access, which provides interface(s) that enable the CSC:business manager in selecting, purchasing and accounting for cloud services;

— administrator access, which provides interface(s) that enable the CSC:cloud service administrator to manage user identity and access, monitor activity and usage and manage faults for cloud services;

— development access, which provides interface(s) that enable the CSN:cloud service partner to access to a set of capabilities within the provider's system that supports the development, test and maintenance of cloud service implementations.

It is usually possible to connect any two otherwise non-interoperable services and make them work together but this can often involve wasteful and time-consuming one-off procedures. Interoperability, on the other hand, has the goal of making any two or more arbitrary services work together. To achieve this, it is necessary to identify interoperability points provided in the services to be considered, for example, by APIs, calls to methods on remote objects, REST requests, etc. and between which clearly defined activities can be successfully completed. In order for a cloud service to be considered as interoperable, all required activities of all necessary CSC sub-roles (as indicated in ISO/IEC 17789:2014, 8.2.2) need to be supported.

A practical approach to interoperability involves the CSC identifying the activities that they need to support for each required CSC sub-role, ISO/IEC 17789:2014, Figure 8-3 and considering what functional components are used to support them, the user function, business function and administrator function shown in ISO/IEC 17789:2014, Figure 9-2.

For example, an on-premises end-user application that uses a cloud service supports the “use cloud service” activity and represents the user function in ISO/IEC 17789:2014, Figure 9-2. Cloud service administrators, cloud service business managers and cloud service cloud service integrators also engage in various activities such as monitor service, request audit report and connect ICT systems to cloud service (shown in ISO/IEC 17789:2014, Figure 8-3), to support the use of the cloud service. For interoperability to be successful, the identified activities need to be supported by business function and administrator function shown in ISO/IEC 17789:2014, Figure 9-2. Each of these functions interacts with interfaces of the cloud service offered by the access layer.

It is necessary that each of the required activities are supported in order to achieve full interoperability in cloud computing.

Interoperability relates to the cloud service customer applications that implement each of the user function and administrator function in the user layer as described in ISO/IEC 17789:2014, 9.2.1 and to how each of them interacts with the relevant interfaces offered by the access layer of the cloud service.

Cloud service interoperability also relates to functional components relating to the CSN, in particular, the cloud service developer role. As shown in Figure 11, the developer environment component interacts with the development access component using the development interface(s) offered by the development access component. The CSN:cloud service developer performs activities including the development, test and maintenance of cloud service implementations.

There are a number of possibilities for how the cloud service customer components are provided. In some cases, they can be an application provided by the CSP, for example, as a web application or as an app for a smart phone or tablet, typically operating through one or more web interfaces to the cloud service. More commonly, the CSC component is an application that is built, developed or purchased by the CSC and that interacts via APIs offered by the components in the access layer. Interoperability considerations are clearly very different between these examples. CSP provided customer components are going to interoperate well with the cloud services they are designed for, but they may not integrate well with other applications and systems used by the customer, including other cloud services. CSC provided customer components might integrate well with other applications and systems used by the customer, but they might or might not interoperate with the cloud service access components.

7.1.2 Transport interoperability

Transport interoperability means the commonality of the communication infrastructure established to exchange data between systems as described in 5.2.1.2. One of the key characteristics of cloud computing, broad network access assumes a communication infrastructure that is common between CSCs and CSPs, providing the foundation for transport interoperability. Such communication

infrastructure is often public, such as the internet, but could also be private. Most communication technologies are based on mature technical specifications and documents, and any of these could be used to support this facet of cloud interoperability.

Interoperability for user, business and administration functions for infrastructure, platform and application capabilities types all rely on a communication infrastructure. Typically, these functions are supported by the same infrastructure but need not be. An example would be the use of the public internet for user functions and a private network for the administration functions.

The essence of transport interoperability is that the CSC and CSP use the same communication infrastructure. If this is not the case, then the CSC is likely to be forced to adapt their systems and applications in some way. The CSC might alter their systems to adopt the communication infrastructure of the CSP or the CSC might install some form of communications adapter such as an ESB to map between the customer systems and the cloud service. These considerations apply both when connecting CSC's systems to a cloud service and also when switching from one cloud service to another cloud service.

7.1.3 Syntactic interoperability

Syntactic interoperability is the ability of two or more systems or services to understand the structure of exchanged information as discussed in 5.2.1.3. Syntactic interoperability concerns itself with the encoding of data for transmission over a communications infrastructure. To be interoperable, the encoding needs to be mutually understood by each system. The ability of a receiving system to decode and examine exchanged data does not guarantee the content is usable by the recipient. For that, the semantic data, behavioural and policy facets of interoperability need to be considered.

Numerous concrete syntaxes, including standardized ones, have been defined to facilitate the encoding and subsequent communication of data between systems. Examples include XML, JSON and ASN.1. The choice of encoding syntax depends on the capabilities of the cloud service and can be different for each of the service, business, and administrative interfaces.

From a syntactic interoperability facet perspective, there is very little difference between the service, business and administrator interfaces and between cloud services based on the different cloud service capabilities types. Each cloud service has a description and definition of its interfaces, which are made available to the participating systems to facilitate a shared understanding. For interoperability to take place successfully with respect to this facet, the interface definitions need to specify the encoding syntaxes required to access their functions and these syntaxes need to be used by the relevant customer components. Use of the encoding syntaxes is necessary but not sufficient to achieve interoperability.

7.1.4 Semantic data interoperability

Semantic data interoperability is the ability of the systems exchanging information to understand the meaning of the data model within the context of a subject area as described in 5.2.1.4.

For all functional interfaces (service interfaces, business interfaces and administrator interfaces) and for cloud services of all capabilities types, successful semantic data interoperability relies on a common understanding of the meaning of the data being exchanged. Thus, semantic data is concerned with the meaning of the information being transferred. Semantic data interoperability for cloud services requires the mutual understanding between the CSC's systems and cloud services of the meaning of the data being exchanged through the service's interfaces. An interface data model needs to be provided to describe the operations and data parameters required to use the service interface.

7.1.5 Behavioural interoperability

7.1.5.1 General

Behavioural interoperability, where the results of the use of a service matches the expected outcome, is described in 5.2.1.5. It includes external and internal behaviour. External behaviour includes all

information visible to external parties outside the interoperating systems. Behaviour is typically described in terms of:

a) internal behaviour that includes:

   1) intended use as described by the CSP;

   2) invariants maintained by the services during the operation;

b) external behaviour that includes:

   1) expected results as understood by the CSC and CSP;

   2) pre-conditions to be fulfilled prior to the operation;

   3) post-conditions reflecting the state of the service upon completion of the operation;

   4) orchestration and dependencies with respect to other services required for correct service operation;

   5) response to management operations.

7.1.5.2 Behavioural interoperability for infrastructure capabilities type cloud services

Behavioural interoperability for a cloud service of infrastructure capabilities type applies to interfaces that deploy and manage customer applications, allocate and manage storage capabilities and configure and operate networking capabilities. It requires that the resultant state match the requested state within the constraints of the service level agreement.

Behavioural interoperability may use formal code-level annotations, e.g. pre/post-conditions, invariants, assertions and milestones. These allow expression of the intended behaviour.

When considering a cloud service of infrastructure capabilities type, behavioural interoperability applies to the APIs which deploy the customer applications, move cloud service customer data and setup network capabilities in the service environment and manage the applications, data and networking within the service. For example, the application of storage for cloud service customer data.

7.1.5.3 Behavioural interoperability for platform capabilities type cloud services

Behavioural interoperability for a cloud service of platform capabilities type applies to interfaces that deploy and manage customer applications. It requires that the resultant state match the requested state within the constraints of the service level agreement.

When considering a cloud service of platform capabilities type, behavioural interoperability applies to the APIs that deploy customer applications and data into the service environment and manage the applications and data within the service and also to the combinations of run-time environments and services.

7.1.5.4 Behavioural interoperability for application capabilities type cloud services

Behavioural interoperability for a cloud service of application capabilities type applies to the interfaces that invoke the functionality of the service, plus those interfaces used to administer the service and perform business activities in relation to the service. These include both the human user and the programmatic interfaces. The pre- and post-conditions for operations and other constraints on the operations' behaviour are needed for interoperability.

7.1.6 Policy interoperability

Policy interoperability is defined as the ability of two or more systems to interoperate while complying with the legal, organizational and policy frameworks applicable to the participating systems as

described in 5.2.1.6. This facet concerns regulations, policy terms and conditions and organizational policies amongst the stakeholders, in any combination of interoperability between CSP, CSC and CSN.

One important area of policy interoperability are regulations which affect the jurisdiction in which data can be stored and in which processing can take place. Such regulations can affect both the CSC and the CSP and can dictate which countries or regions are acceptable for storage and processing. This is particularly the case for cloud services involved in the storage and processing of certain types of data such as PII, health data, financial data and government data. Storage and processing can only take place where cloud services are located in permitted jurisdictions. Sometimes, this can include not only the location of the data centre where storage and processing take place, but also affects the location of the staff who operate the data centre, where they have access to the data and processing systems. This sometimes extends further to limit the nationality of the staff involved. Regulations of this type can severely restrict a set of cloud services that can be used by the CSC for the data and the processing covered by the regulations.

The use of applications and services on various mobile platforms could be restricted by local geographic constraints imposed by local government jurisdictions. These governmental jurisdictions influence what functionality can be offered by the CSP on the mobile platforms for cases when the customer moves from one jurisdiction to another.

Policy interoperability can influence the choice by the cloud service customer of cloud service and/or the data centre location used for the cloud service.

Regulations which apply to the CSP can also influence interoperability. In some jurisdictions, organizations responsible for national security can have the right of access to data stored within the jurisdiction. Such access might be unacceptable or even forbidden by the jurisdiction applying to the CSC or the CSP, except under international cooperation treaties.

Policy interoperability includes regulations or policies which relate to specific capabilities of the cloud services. One of the principal capabilities of concern is security. There can be specific security requirements based on the nature of the data and the processing involved, for example, where credit card information is concerned, the cloud service can be required to meet the requirements of the Payment Card Industry Data Security Standard (PCI-DSS)[22]. There can be more general security requirements demanded by policies of the CSC, such as a requirement for the cloud service to be certified against a standard such as ISO/IEC 27001 or ISO/IEC 27017.

7.1.7 Interoperability with connected devices consuming cloud services of application capabilities type

Increasingly, the users of cloud services are applications that run on connected/mobile devices, e.g. social media, over-the-top communications apps, workplace and productivity apps, etc. There are scenarios involving interoperability of connected/mobile device applications with cloud services that provide the back-end of those applications' functionality. Furthermore, application marketplaces (as defined in ISO/IEC 19944) are themselves cloud services of application capabilities type that interoperate with connected device applications. Device users and application developers benefit from interoperability where applications of any device platform can interoperate with cloud services offered by other CSPs.

— Examples of connected devices are mobile devices, e.g. smartphones and tablets and Internet of Things (IoT) devices.

— Cloud service customer data (as defined in ISO/IEC 17788) can be cached, accessed and manipulated via users' connected devices, e.g. photos, text, instant messages, emails, notes, etc. accessed by mobile devices such as smartphones and tablets. The caching, manipulating or otherwise accessing cloud service customer data that takes place on the connected devices is fundamentally an interoperability interaction.

— Users might expect their cloud service customer data to be accessible as they move from one device to another, independent of the device platform and the corresponding device application

marketplace (as defined in ISO/IEC 19944). This means interoperability between connected device applications and cloud services containing the cloud service customer data is needed.

— Connected devices can capture and upload cloud service customer data to various cloud services using applications running on the device. Such cloud services are of "application capabilities type". The cloud service customer data can also be available to other applications on the original connected device. These scenarios represent interoperability between applications and device platforms of the connected devices on one end, and cloud services on the other. Such a degree of interoperability can help prevent platform or provider lock-in.

— Users benefit where the applications available from the device application marketplace to interoperate not just with device platform cloud services, but also with alternative cloud services supporting functionality needed by those applications.

8 Cloud data portability

8.1 Cloud data portability types

Table 5 describes the types of cloud data portability scenarios based on cloud capabilities types for each facet of cloud data portability (see 5.2.2). Each cell at the intersection of the rows and columns of the table indicates the clause where there is a description of what needs to be taken into consideration for data portability between CSPs or between a CSP and CSC.

Table 5 — Cloud data portability types

Cloud data portability facets	**Cloud capabilities types**		
	Infrastructure	**Platform**	**Application**
Data syntactic (5.2.2.2)	8.2.2	8.2.3	8.2.4
Data semantic (5.2.2.3)	8.3.2	8.3.3	8.3.4
Data policy (5.2.2.4)	8.4		

Where the concerns of a cloud data portability type are the same or very similar to another type, they have been collected into the same subclause.

8.2 Data syntactic portability

8.2.1 General

Data syntactic portability concerns itself with the format and encoding of data into suitable data artefacts that are capable of being transmitted to and decoded by another system, such as a cloud service or a CSC's system. To be portable, each artefact needs to be encoded using an interchange format that can be decoded by the target system, which may involve transformation between formats. Typically, the artefact is labelled with an encoding type, which might be indicated by a file extension or MIME type. Examples are office document files, pictures and photos, comma separated values (CSV), and XML files. A data artefact might itself contain other data artefacts encoded using the same or a different format. Examples are a Zip file containing other data artefacts and a hierarchical file system containing directories of data artefacts.

Where the target cloud service uses a data syntax that is different from that used in the source system (whether an on-premises system or in another cloud service), data portability is still possible, but it generally requires a transformation step to take place. For example, the source might be in XML, but the target requires JSON. Such a transformation is generally possible and in some cases, can be performed using widely available tools. Other cases can require a transformation tool to be built or configured.

A system receiving data artefacts might be able to decode and examine the contents, but this does not guarantee the content is understood in the context of the recipient. For that, the semantic facet of data portability needs to be considered.

A cloud service can internally store and represent data how it sees fit, determined by implementation decisions. Multiple cloud services offering the same functionality through identical, possibly standardized, interfaces are under no obligation to internally store and structure data in the same way. However, by using structured, commonly used machine-readable data interchange formats, data has a higher probability of being portable between cloud services.

For the case of physical media used to transport data, syntactic data portability is needed if the physical media are to be consumed properly.

8.2.2 Data syntactic portability for infrastructure capabilities type cloud services

A cloud service that supports the infrastructure capabilities type, such as IaaS, allows a CSC to provision and use processing, storage or networking resources. For example, this enables a CSC to upload virtual machine images and data objects so that applications that run inside an instantiated virtual machine can provide services to the CSC's users.

The data artefacts thus relate to virtual machine images, file systems, content stores, databases, etc. Common packaging formats are particularly helpful for infrastructure data portability. Examples are Open Virtualisation Format (OVF), Zip and tar. The detailed format of data objects contained within the packaging is often of little concern to the infrastructure capabilities type cloud service. For example, a file or data object can be stored on storage infrastructure within a cloud service without the cloud service having to know anything about the syntax and format of the data within the file or data object. It is likely that the content of such a file is only accessible by application components belonging to the CSC.

The infrastructure data artefacts and the packaging and encoding formats that are supported by a cloud service are typically documented in the cloud service agreement. A cloud service is expected to deliver and receive infrastructure artefacts in the one of the supported formats. However, for operational reasons, a CSC might not receive back infrastructure data artefacts in the format in which they were originally supplied to the cloud service.

Data portability has a close relationship to application portability when discussing syntactic aspects of infrastructure capabilities cloud services, since aside from packaging, the artefacts under consideration are the same. For example, a virtual machine image could be considered a data artefact and an application code artefact with respect to syntactic data portability for infrastructure capabilities type cloud services. Application syntactic portability for infrastructure capabilities type cloud services is covered in 9.3.2.2 and it is advisable to consider it in parallel with this clause.

8.2.3 Data syntactic portability for platform capabilities type cloud services

A cloud service that supports the platform capabilities type, such as PaaS, allows a CSC to deploy, manage and run customer-created or customer-acquired applications using one or more programming languages and one or more supported execution environments.

This enables a CSC to upload applications, data stores and related artefacts and for them to be on the target cloud service, so that the applications can be used by the CSC's users. The data artefacts thus relate to applications, either source code or compiled executables, the data stores that contain cloud service customer data, such as databases or file systems and metadata to help assemble the artefacts for execution. While many of the artefacts themselves are environment or platform-specific and thus the formats are dictated by the platform, common packaging formats are particularly helpful for platform data portability.

The platform data artefacts and the packaging and encoding formats that are supported by a given cloud service are typically documented in the cloud service agreement. A cloud service is expected to deliver and receive supported platform artefacts in the supported formats, however, for operational reasons, a CSC might not receive back platform artefacts in the format which were originally supplied to the cloud service.

The syntax and format of data objects transmitted to a platform capabilities cloud service might be dictated by the functionality of the cloud service. Platform capabilities cloud services can include basic data storage facilities such as file or object stores; these can generally handle files and objects containing data using any syntax and format. Other capabilities might include database services, which can demand that supplied data is in very specific syntax and format. Capabilities such as runtime platforms can also demand specific files with specific content, held in specific directory structures.

The architecture of the application dictates the options for data portability for platform capabilities types. Data that is intimately bound to the application might not be possible to port without porting the entirety of the application. This scenario is very similar to application syntactic portability and for this reason, it is necessary to consider 9.4 in parallel with this clause. In other cases, it may be possible to move the data artefacts containing the cloud service customer data independently of moving the application.

8.2.4 Data syntactic portability for application capabilities type cloud services

A cloud service that supports an application capabilities type, such as SaaS, allows a CSC and its users to use applications provided by the CSP. Data portability for application capabilities type cloud service involves the uploading, either bulk, piecemeal or both, of cloud service customer data for use by the cloud services. A CSP should declare what data artefacts are supported and what packaging formats are acceptable.

All cloud service customer data that are uploaded by a CSC should be retrievable and are expected to be in one of the advertised formats.

The data artefacts are application dependent and will vary greatly given the variety of diverse applications. Even applications that deal with a similar problem domain, such as CRM, can have wide variation in the data artefacts that reflect different features offered by different cloud services. The many different industry verticals for which cloud services are offered, such as finance, healthcare, retail and HR, means that there is often little in common between the data artefacts from cloud service to cloud service.

Unlike infrastructure and platform data artefacts, application data formats are not necessarily in commonly understood formats, such as MIME types for example. Even if a common format is used, there may be extra syntactic rules, structural and grammatical, defined that further elaborate on the structure of the contents of the data artefact. Application-specific data formats are typically expressed as a data interchange format defined in languages that can convey more structure such as XML schemas, Schematron (see ISO/IEC 19757-3) and UML. For data artefacts to be meaningful when porting data between a CSC's system and a cloud service and between cloud services, a mutual understanding of application-specific data interchange formats is essential.

Even if common syntax formats are used, such as ASCII text files or CSV, those formats may not be able to convey all the semantic information required by the cloud service. For example, representing an organization's employee directory data in an unstructured ASCII file may lose information as not all the relationships between people and departments can be expressed; this is a semantic data loss due to using a format that cannot adequately represent the semantics.

When a CSC wants to switch to a different cloud service, the CSC needs to be able to take their data artefacts from the original cloud service and move them to the target cloud service. The switching cost will include exporting, mapping and importing the data into the target application capabilities type cloud service and that cost is a function of how well the data models and interchange formats of the two cloud services line up. Standardizing these interchange formats greatly reduces switching costs. However, CSPs might not support exactly the same type of cloud service with identical data artefacts. Therefore, not all the data artefacts exported from one provider might be required or supported by another. Thus, there may be a loss of fidelity when porting the data. From the application standpoint, it is the data artefacts and formats that benefit from standardization rather than identically functioning and featured cloud services.

8.3 Data semantic portability

8.3.1 General

Data semantic portability is data portability such that the meaning of the data model is understood within the context of a subject area by the target system. A (logical) data model, sometimes called a meta-model, expresses the data items, their names and their attributes, logical structures, relationships and other constraints that are required.

Semantics concerns itself with concepts within a domain, their properties, terminology and vocabulary and the structural relationships between them. Semantics also concerns itself with the correctness, validity and completeness of data. An example of correctness is determining whether a zip/postcode is valid for the address provided. An example of completeness is data that says there are 100 employees in a department, yet only 50 employee records are provided.

Data models in a cloud computing context are dictated by the types of cloud service offering as outlined in the following subclauses.

8.3.2 Data semantic portability for infrastructure capabilities type cloud services

For infrastructure capabilities type cloud services, the domain of interest relating to data concerns compute, storage and networking metadata, typically to enable the migration of VMs and workload data.

The data artefacts thus relate to virtual machine images, file systems, content stores, databases, etc. It is typical of infrastructure capabilities type cloud services not to have concerns in relation to the semantic content of these data artefacts, since the data artefacts are processed by applications belonging to the CSC and not by the cloud service itself. The only case where the semantics of the artefacts are of direct concern to the cloud service relates to program images intended for execution in a compute capability cloud service, which is addressed under application portability in 9.3.

8.3.3 Semantic data portability for platform capabilities type cloud services

For platform capabilities type cloud services, the data models concern applications intended to be installed. This includes artefacts related to application code, either source code or compiled executables and metadata to help assemble the artefacts for execution. There is also cloud service customer data held in data stores, such as databases or file systems and object stores.

In most cases, the semantics of the cloud service customer data is not of direct concern to the cloud service itself since the data is typically processed by application code belonging to the CSC. However, platform capabilities type cloud services can include sets of services and some of these services might be used by the cloud service customer applications. In this case, the services can end up processing some of the cloud service customer data and if this is so, the semantics of the cloud service customer data matters greatly. It is necessary for the semantics of the data to match that expected by the relevant services or else unexpected results are likely to occur. It is likely to be necessary for the CSC to address any differences between the semantic expectations of the services used and the semantics of the original cloud service customer data.

For application code artefacts, semantic considerations are explored under cloud application portability for platforms in 9.4.

8.3.4 Data semantic portability for application capabilities type cloud services

In the case of application capabilities type cloud services, the application code of the cloud service operates directly on cloud service customer data. As described in 8.2.4, the domain concepts and data model are dictated by the application itself. It is necessary for the CSC to have a thorough understanding of the semantics of the data used by the cloud service application and to ensure that cloud service customer data supplied to the application has the necessary semantics. A data model of an application domain is defined by exploring the concepts within the domain, their properties, terminology and vocabulary and the structural relationships between concepts. When examining and comparing data

artefacts from non-cloud applications and/or from cloud services with a view to porting data between them, Table 6 contains a non-exhaustive list of considerations that apply.

Table 6 — Example considerations for alignment of application semantics

Topic	Considerations
What are the domain concepts covered by an application?	— A cloud service that offers email services has the concepts of email address, sender, recipient, subject, body and attachments, together with the ability to organize emails into folders. — A photo sharing cloud service has the concepts of photos and photo collections or albums. — A human resource cloud service has departments, employees and reporting hierarchies.
When comparing two applications, are the domain concepts similar to each other?	— Email services will be expected to have the same concept of an email and an email address, however, some services do not offer a traditional folder/sub-folder way of organizing emails. Therefore, this information may get lost unless the concepts are mapped, such as folders mapped to labels.
When comparing two applications, is the vocabulary really equivalent in meaning?	— Is a picture identical to a photo? Is a record in a healthcare cloud service the same as a record in a music service?
When comparing the models of two applications, are the terms used for each concept equivalent? Are they synonyms or are terms being used for very different concepts?	— If two cloud-based photo sharing services use the term photo, is that term being used for exactly the same concept, e.g. is the metadata, location, ISO sensitivity setting, etc. included or not? — It is possible for different photo sharing cloud services to use different terms for the same concept. For example, one service uses the term “photo”, while the other uses the term “picture”.
When comparing the models of application, are relationships between the concepts equivalent or different?	— One cloud-based HCM service could have the concept of a dotted line relationship between two employees, to express a non-managerial relationship; whereas another cloud-based HCM service may only allow direct managerial reporting relationships to be expressed. — This could lead to loss of information when porting data between applications that differ in this way.
When comparing the models of application, are classifications/typologies equivalent?	— Concepts can be classified into various groupings based on the domain.
When comparing the models of application, are constraints and rules equivalent?	— Items in a data model can have restrictions placed by constraints and domain rules; the constraints can differ between different cloud services.

Concepts, vocabulary and relationship between concepts are vital to the understanding of a domain model and in understanding, any differences between applications, whether in non-cloud or in cloud services. If semantic data portability is to occur between applications, there is a need to have a mutual understating of the domain model(s). The use of semantic languages, such as OWL and Resource Description Framework (RDF) and modelling notations such as UML and Business Process Management Language (BPML), are tools that can be used to document a shared understanding. It is common for modelling tools that support notations like UML and BPML to define interchange formats that can be used at the syntactic data portability layer. As well as generic languages to express domain models, various vertical application areas have defined specific semantic models, e.g. life sciences[2)].

Where the semantic models of the source and the target of data portability differ, then there is a need to perform some kind of semantic mapping between the source data and the target data. This mapping process might be straightforward or highly complex depending on the nature of the differences between the semantic models.

2) https://www.w3.org/wiki/SemanticWebForLifeSciences

8.4 Data policy portability

Data policy portability is defined as the ability to transfer data between a source and a target while complying within the legal, organizational, and policy frameworks applicable to the source and target. This includes regulations on data locality, rights to access, use and share data, and mutual responsibilities with respect to security and privacy between a CSP and a CSC. It also includes any enterprise policies, particularly of the CSC.

Table 7 is an exemplary list of topics to be considered.

Table 7 — Example considerations for data policy portability

Topics	Considerations
Data use rights	a) Verify that the proposed data port complies with any license restrictions that may be in place, such as: 1) audio/visual media licensed from a commercial content provider; 2) mapping or other commercial data licensed from a data provider.
Service use rights	a) Verify that the target cloud service to be used meets legal requirements for handling the data being ported, such as: 1) Certified for handling health data; 2) Meets required privacy standards or regulations, e.g. ISO/IEC 27001 with ISO/IEC 27018; 3) Meets government procurement rules.
Data location	a) Verify that the target cloud service stores data in appropriate jurisdiction(s) to meet organization/customer/regulatory requirements and obligations.
Processing location	a) Verify that the target service processes the data in appropriate jurisdiction(s) to meet company/customer requirements and obligations.
Third party processing	a) Verify that any third-party processing is well-understood and that the third party also meets the requirements and obligations, and will continue to do so.
Jurisdiction	a) Verify whether the porting of data changes the applicable jurisdiction(s) from that of the source system and investigate any legal consequences of any such change.
Contract	a) Take appropriate legal advice. b) Keep and maintain appropriate audit trails. c) Understand any contractual limitations that exist in relation to the data being ported.
Security and privacy	a) Verify that the target environment provides the necessary security and privacy controls to meet the policies of the CSC with respect to the data being ported.

These considerations in Table 7 are applicable across cloud services of all capabilities types.

8.5 Considerations for cloud data portability of "cloud service derived data"

Cloud service derived data is the class of data objects under CSP control that are derived as a result of interaction with the cloud service by the cloud service customer (see ISO/IEC 17788). An example of cloud service derived data are logs of user interactions with the cloud service.

There can be many types of cloud service derived data and not all of those types are portable. One main reason for classifying this type of data is to emphasize that it may have different portability characteristics from that of cloud service customer data and cloud service provider data. A pair of CSC and CSP can agree on data portability requirements in any way that suits their needs, however, it is typically the case that CSPs do not make all cloud service derived data portable for a number of reasons, both technical and business related. Examples of such reasons are the data not being relevant to the CSC or the data containing derived data from multiple CSCs and not being possible to be shared while respecting the privacy and confidentiality of all CSCs. Table 8 contains some considerations.

Table 8 — Example considerations for cloud data portability of cloud service derived data

Topics	Considerations
Extract and erase	— Unlike cloud service customer data, which is assumed to be portable and erasable under the CSC's control, the ability to extract cloud service derived data from the system for use by the CSC or to enable erasure of some cloud service derived data by the CSC is likely to need careful control and is subject to agreement between the CSC and CSP.
Regulations	— Regulations can apply to some types of cloud service derived data. For example, some types of log information might have to be retained for a specified period and cannot be deleted.
Categorization	— The detailed categorization of cloud service derived data in the taxonomy defined in ISO/IEC 19944 is intended to support the definition of the cloud service agreement between the CSC and CSP. This data categorization can be utilized when CSPs and CSCs engage in defining the portability requirements of cloud service derived data. In such cases, the agreement may reference specific sub-types of derived data in the portability discussions and agreements.
Scope	— Cloud service derived data has potential meaning outside the cloud service (otherwise, it would fall under the categorization of cloud service provider data) and the CSC may wish to access some data categories of derived data, or to request erasure of some categories of derived data.
Other requirements	— Other cloud computing standards, notably ISO/IEC 27017 and ISO/IEC 27018, contain requirements in relation to the availability to the CSC of certain types of cloud service derived data.
Aggregation	— Cloud service derived data collected by a CSP is sometimes aggregated with that of other CSCs and in many cases, is de-identified to remove PII. In such circumstances, providing the data records specific to a CSC and its users is technically challenging and adds risk to the confidentiality of other tenants.
Data minimization	— Making some types of cloud service derived data available to the CSC could interfere with data minimization policies designed to protect privacy and confidentiality. These policies dictate shortened data retention periods, de-identification of data and erasure or masking of records not needed to provide the cloud service. Removing these policies across all types of cloud service derived data to permit future access or erasure by CSCs is often unacceptable.
Challenges	— There are circumstances, such as CSC access to log files, where the provision of certain categories of cloud service derived data specific to the CSC is an important requirement. However, the technical challenges and risk of confidentiality and privacy to other tenants means provision or erasure of these types of cloud service derived data that needs to be explicitly defined and carefully controlled.

Table 8 *(continued)*

Topics	Considerations
Analytics	— CSPs could run data analytics algorithms on cloud service customer data. The results could also be combined with cloud service derived data collected as the user interacts with the capabilities of the cloud service(s). Such a combination should still be treated as cloud service derived data but it might have lost relevancy to a given CSC. Such a combination could generate cloud service derived data that can be the basis for offering additional new insights to the CSCs and their users about their data via new features or improved capabilities of the cloud service(s). In many such cases, the cloud service derived data is used to create the new and improved capabilities and feature set of the cloud service, but by itself it is unlikely be useful to the CSC. Therefore, these categories of cloud service derived data might not be portable.
Graph data	— Some applications develop social graph data that relate to cloud service users and other artefacts that are stored in the corresponding cloud service. Such data are unlikely to be portable as they are highly cloud service implementation specific and combine cloud service customer data and cloud service derived data from multiple users and other sources. The portions of the data that are meaningful outside the social graph and are part of cloud service customer data are normally available to the CSC.
PII	— Care needs to be taken not to compromise PII of a data subject as well as that of other associated data subjects when porting cloud service derived data.

Given the above considerations, the portability of cloud service derived data is typically considered on a case by case basis. Ultimately, the decision is made by the CSCs and CSPs and is captured in the cloud service agreement between the two parties. Once the types of cloud service derived data artefacts available to the CSC are agreed, the issues regarding data portability are as described above.

9 Cloud application portability

9.1 Cloud application portability types

Table 9 describes the types of cloud application portability based on cloud capabilities types for each facet of application portability. Each cell at the intersection of the rows and columns of the table is a cloud application portability type and it is described in the indicated subclause in this document. Each clause describes what needs to be considered for the porting of the application. Notice that in some cases, there is a need to differentiate between application portability from non-cloud deployments to cloud services as opposed to cloud service to cloud service scenarios.

Table 9 — Cloud application portability types

Cloud application portability facets	Cloud capabilities types					
	Infrastructure		Platform		Application	
	Non-cloud to cloud	Cloud to cloud	Non-cloud to cloud	Cloud to cloud	Non-cloud to cloud	Cloud to cloud
Application syntactic (5.2.3.2)	9.3.1.2	9.3.2.2	9.4.1.2	9.4.2.2	9.5	
Application instruction (5.2.3.3)	9.3.1.3	9.3.2.3	9.4.1.3	9.4.2.3		
Application metadata (5.2.3.4)	9.3.1.4	9.3.2.4	9.4.1.4	9.4.2.4		
Application behaviour (5.2.3.5)	9.3.1.5	9.3.2.5	9.4.1.5	9.4.2.5		
Application policy (5.2.3.6)	9.3.1.6	9.3.2.6	9.4.1.6	9.4.2.6	9.6	

Cloud application portability is the ability to migrate an application from one cloud service to another cloud service or between a CSC's system and a cloud service. For the purposes of this document, the application includes the collection of instructions and/or internal data needed to implement that application.

"Internal data" in the above description references the data tightly associated with the software code/instruction, according to the design of software. Configuration or settings data would be an example of such internal data. It is in contrast to external data, for example, a Joint Photographic Experts Group (JPEG) photo file that a photo editing app would open is considered external data, but the workflow settings for that same application would be internal data.

9.2 Considerations for cloud application portability

When porting applications from one environment to another, there are some high-level considerations as shown in Table 10.

Table 10 — Example considerations for cloud application portability

Topics	Considerations
Understand the overall or business objectives for porting the application.	— In some cases, the move is for technical reasons such as lack of support in the old system, but in most cases the move likely includes business reasons such as the provision of new business capabilities that the new environment provides. It is also likely that there are many objectives in moving to or from a cloud service and these objectives inform the decisions that are made in this port. For example, if the application is being ported to a cloud service to support new business processes, e.g. mobile device support, machine learning capabilities etc., it is likely that some re-write of the application is necessary to achieve these objectives, although such changes go beyond the scope of application portability.
Define what is meant by the "application".	— Business or end users may define the application by the business functionality that is provided to them. Whereas the technical definition of the application may encompass much more business functionality, as well as many related services such as security, access control or other applications that provide data or services that are essential for the operation of the application. Defining the application artefacts and the associated service dependencies, as defined in 6.2, is an essential first step when porting the application to a target environment.
Decompose the application into its elements.	— Once the application artefacts and its associated dependencies are identified, it is possible to decompose them into a set of elements that each need appropriate handling as part of the application porting process. Depending on the nature of the artefacts and the target environment, this step might be difficult or very easy. For example, if an application consists of three virtual machine instances and it is only the machine instances that are being ported, the decomposition could be very straightforward.
Understand the application dependencies.	— The complexity of the application itself can be an issue, but of greater consideration is its set of dependencies, including the usage of other services and applications. Some applications cannot be easily removed from their complex set of dependencies which can include services such as security and access, data which is shared by other applications, other applications themselves and many other services which were assumed to be available when the application was originally conceived such as internal networks, console screens, printers and so on. Decisions need to be made about what to do with each of the dependencies, for example, whether to port them as-is or whether to replace them with equivalent capabilities available from the cloud service provider.
Port, build or buy.	— A big reason to move to a cloud platform service is to use those services that the platform provides. For many CSPs, additional services are being added continually, so it is likely to make economic sense to alter the application to take advantage of these services to reduce the application maintenance required and to quickly avail the application of these new cloud services. In many cases, services such as mapping, mobile device support, big data analysis and others can provide business benefit and be very easily added to the application. Some of these cloud platform services, such as identity and access services, are best not considered in just the scope of one application, but in the scope of the overall business functionality it provides. It is important to distinguish between the porting of existing application capabilities and the extending and modifying of the application to encompass new or extended capabilities. Application portability strictly applies only to the first of these processes, although it is often the case that in porting to a new and richer environment, advantage is taken to extend the application at the same time.

The application portability facets describe the requirements that all need to be met for application portability to be successful. Each of these facets has a number of items that may need to be considered in particular portability scenarios. An explanation of these considerations is outlined in the Table 11.

The considerations presented in Table 11 are applicable to the cloud application portability topics in Clause 9. Further details specific to each sub-topic is discussed in the corresponding subclause.

Table 11 — Example considerations for application portability facets

Application portability facets	Example considerations when porting applications
Application syntactic	— **Packaging**. Applications and their associated metadata and instructions need to be packaged in a format that can be read in the target environment. — **Delivery**. The package needs to be delivered to the target environment, understood on receipt and unpacked in a manner that makes the result manageable by all the necessary actors, which include developers, testers and most likely environment services to support the package(s). — **Security**. Not only might the delivery be encrypted to assist with securing the package, other security measures including access controls need to be considered in porting the application.
Application instructions	— **Definition and scope of application stack**. An application is generally made up of many software components, such as a web page interface, a set of business processes, data handling and access control and so on. This is the "application stack". It is important to understand how much of that stack needs to be ported to the new environment and that will depend on many factors, both technical and business related. — **Control of application stack**. "Application portability" means taking the application from one environment and moving it to another. It often involves making some changes as part of the porting process. This needs to be achieved for all the elements of the application stack. Changing code for some elements might be difficult, especially for commercial code with no access to source code. — **Application services**. Even when the application is running in a non-cloud environment, an application can make use of multiple services. Some of these are directly related to the application itself (such as totalling the product lines in an invoice) but others (like measuring the time taken) can be derived from other services within the application. For the application to port successfully, all these services need to be available to the application in its target environment. — **Application supporting services**. The supporting services of an application are not contained within the application itself, but are necessary for it to function correctly. When porting applications, it is necessary to understand how such services are made available to the application in its target environment and also any services that support those services. — **Development and testing services**. Applications are built and tested using tools that are designed to work with a particular environment. If the application is ported to a target environment, it is possible that the tools also need to be ported or to use other alternative tools designed to work with the target environment. — **Application instruction support**. Applications are written in a language. This language could be a common code such as C# or Java or it could be a specification such as BPEL. In either case, the target environment needs to support whatever instructions are used.
Application metadata	— **Definition and scope of metadata**. Data that is not the subject or focus of the application, but is necessary to describe its resource requirements or operation is considered application metadata. When porting the application, it is necessary to define this metadata and what part of the metadata is within the scope of the port. — **Static, environment and global variables**. The application runs in an environment that offers many environmental resources. These can be as varied as access to the current date and time to the status of other applications such as "batch run complete" or "user language is French". It is important to consider which of these variables are to be ported, reconstituted or redesigned in the target environment. — **Metadata maintenance**. In some cases, the maintenance (including updating and improving) of metadata is part of the application development process. In many cases, maintenance is done outside the application. It can be provided by other applications, by the in-house systems or by a CSP. The application developers might need access to change the metadata or to rewrite the metadata for the target environment.

Table 11 *(continued)*

Application portability facets	Example considerations when porting applications
Application behaviour	— **Application input/output test cases**. Because the nature of an application can change when it is ported to different environments, a way to assess whether it is compatible is to construct test cases with given inputs that produce expected outputs. This technique can remove the technical complexities and implementation details so as to focus on the results of the application. — **Application technical test cases**. These test cases focus on the technical expectations of the application, including response times, security and resilience. — **Business process flows**. An application is but one part of a large business process. In some cases, porting the application to another environment can result in a slightly different application. In many cases, these differences are by design and are often the reason for the application port. In other cases, they can be an unwanted by product. In either case, it can be more advantageous to change parts of the business process that are outside the application itself, either to improve the business itself or to reduce the costs involved in changing the application.
Policy, legal and organization	— **Application use rights**. These rights can be granted by law, regulation and contract or just by the organization but they give the user/group/organization the right to use the application. They could also include software licensing. The key question is whether the application use rights permit the application to be run in the target environment. — **Service use rights**. Similar to application use rights, service use rights allow cloud services to be used. In many cases, an "application" can be thought of as a collection of services so the rights applying to those services can be "finer grained" than application rights. — **Data location**. The actual location of the data (or metadata) can be important for legal or other reasons. — **Processing location**. In some cases, where the processing takes place is an important issue. — **Third party processing**. With non-cloud applications, the organization itself processes data through its applications. When the application is run in a cloud service, that processing can be done by the CSP. The organization is likely to be the data controller for any PII being processed, while the CSP is likely to be a third party data processor. — **Jurisdiction**. Laws and regulations are generally scoped within a physical jurisdiction. Porting an application to a cloud service can also move the jurisdiction or spread the application across multiple jurisdictions. — **Security and privacy**. It is necessary to verify that the target environment provides the necessary security and privacy controls to meet the policies of the cloud service customer with respect to the application being ported. — **Contract**. Contracts between parties can outline special conditions for applications and their portability.

9.3 Application portability for infrastructure capabilities type cloud services

9.3.1 Application portability from non-cloud to cloud service

9.3.1.1 General

The following is a use case describing an example of application portability. An organization moves a three-tier application from a non-cloud data centre to an on-premises cloud system or to a CSP that runs the application in the CSP data centre.

A three-tier application consists of the front-end web server, back-end database and middle-tier business logic that services data requests between web server and the database.

Actors involved in this use case include CSPs, software developers and system administrators/operators.

This use case represents the most common type of web-based application deployed both in enterprises and mid-sized companies. Thus, the three-tier app is often an entry-level point for larger organizations to learn how to migrate gradually from the traditional client/server, web-based model to cloud computing.

Clause 6 offers a model for non-cloud applications that are to be ported to cloud computing environments (see Figure 13). Contrasting to the model described in Figure 14 in the same clause for infrastructure capabilities type of applications is instructive.

One of the common challenges in porting an application from a non-cloud deployment is defining the scope and boundaries of the application, identifying all the application artefacts and also identifying all the service dependencies of the application. Once all the artefacts and dependencies are identified, then it is necessary to define an approach for porting each of them.

The non-cloud application typically has all its dependencies provided by the enterprise environment, while depending on data centre resources such as storage. When such an application is moved to an infrastructure capabilities type cloud service, most if not all of its dependencies are likely to have to port over as well.

The target cloud service might offer capabilities such as virtualization and network services that the ported application can take advantage of. The porting process needs to consider the packaging format used for the existing application binaries, for example, a VM image or a container image and the management APIs to monitor and control the application instances once they are executing in the cloud service. Other than changes needed to move the application from its dependencies on data centre resources to the equivalent resources in the cloud service, the rest of the application ordinarily does not need to be modified, given the same application executables would run on top of the same operating system (albeit inside a VM or container) once ported to the cloud service.

9.3.1.2 Application syntactic portability

Syntactic portability means that the target cloud service needs to understand the format of the application and its related components as transmitted by the CSC. For the fundamental types of resource offered by an infrastructure capabilities type of cloud service, it is typically the case that the target cloud service is capable of dealing with the syntax of the application artefacts.

For this type of portability, the syntactic portability would involve the packaging format for the application executable binaries and its associated configuration data used for deployment into a VM or container. OVF is an example of such a format.

9.3.1.3 Application instruction portability

Application instruction portability to some extent depends on the nature of the application artefacts and the target runtime environment. For example, applications built using interpreted languages or using machine-independent byte codes such as Node.js, Python, Java or C# normally port "as is" as long as the target cloud service supports the runtime. Note that the runtime itself is usually ported to the target cloud service as well as the application instructions artefacts, in the case of an infrastructure capabilities type of cloud service.

Components written in languages that require compilation to machine instructions, such as C or C++, will have been already compiled to a particular target non-cloud environments environment. Recompilation is not normally necessary where the target cloud service supports the same processor type, operating system and runtime libraries for which the component was originally compiled. More significant work might be required if the target cloud service uses a different processor type, e.g. recompilation. If the target cloud service supports a different operating system or even operating system version, more substantial changes to the application code can be required.

9.3.1.4 Application metadata portability

The application metadata includes information about resource requirements, deployment and configuration information and initialization data. It is usually possible to port the application metadata unchanged between the non-cloud system and the infrastructure capabilities type of cloud services given the application is expected to run in the same execution environment as it had in the non-cloud environment, albeit inside a virtualized environment in the infrastructure capabilities type of cloud service.

9.3.1.5 Application behaviour portability

Application behaviour portability is likely to be satisfied in a straightforward way given the application is expected to be unchanged in the cloud environment, even though all of its components are running in one or more virtual compute capabilities in the target cloud service.

Consideration needs to be given to differences between the non-cloud environment and the cloud service environment. The processors available in the cloud service may be different to those used in the non-cloud system, with potential changes in the performance of the application components. Where the application consists of multiple components running as separate processes, there is the need to consider changes to the speed and latency of the communication between those processes. There may also be changes in response times that are noticeable by end-users of the application.

It is advisable to ensure that there is a comprehensive set of test cases that can be used to check the application behaviour before and after the porting of the application to the cloud service.

Porting an application to a cloud service could be done to make the application more available to a larger user base. This might have technical implications that impact issues such as processing and response times, resilience and security and so on.

9.3.1.6 Application policy portability

Some policy considerations relating to cloud application interoperability include:

— Processing location

Even if the application code and its environment is substantially the same, a change of location, e.g. to another country, may affect the rights and obligations that apply to the usage of the application.

— Third party processing

The organization now processing the data may no longer be the original application user or operator.

— Jurisdiction

The jurisdiction in which the application operates may have changed.

— Contract

The application may be utilizing a code that is licensed from another party and such licensing may not allow implementation in the new environment.

In addition, see 9.6.

9.3.2 Application portability from a cloud service to another cloud service

9.3.2.1 General

This use case is similar to porting a non-cloud application to a cloud service except the application is being moved from one cloud service to another cloud service.

A typical example is to move a three-tier application from running on one cloud service to another cloud service. In such scenario, a CSC moves the three-tier application from one cloud infrastructure provider to another. Actors involved include CSPs, software developers and system administrators/operators.

As in 9.3.1, the ability to port the application from the source cloud service to the target cloud service depends on the capabilities to extract the various application artefacts from the source service and to transfer those artefacts into the target service. This requires appropriate interfaces and formats to be supported by the source and target cloud services. It also requires importing and exporting of data as well as cloud service lifecycle management.

Referencing Figure 14, the application artefacts ideally move without modification and the resource requirements are the same in the target cloud service as in the source. For example, the virtualization resources in the target cloud service are ideally similar to those of the source cloud service. The file format for the application artefacts is ideally portable, otherwise, the file format needs to be modified to match that accepted by the target cloud service. Management capability differences between the source and target cloud services could also affect application portability. If the management capabilities of the source and target cloud services are not similar, additional training for the operators, modification of customer management applications interfacing via API and testing of the ported application artefacts are needed to ensure that the application runs correctly in the target cloud service.

Similarly, network resources also need to be similar between the two cloud services. For example, connections to on-premises systems, e.g. VPNs, or to outside systems, e.g. identity federation servers, need to be easily reconfigurable or reusable to keep the effort needed for porting from the source cloud service to the target cloud service low cost and low risk.

Figure 14 shows that application artefacts are moved unchanged (instruction sets, configuration data and data sets) while there may be a group of service dependencies relating to the application that also need to move, some of which may need to be adapted to the target cloud service and others may be supplied by the target CSP. The important factor is whether a given dependency is provided for by the application itself or whether it is provided by the CSP. The dependencies that are implemented by the application itself are ported as is, and the target cloud service has to provide a virtual environment that supports the same artefacts as the source cloud service. However, those dependencies that are provided for by the source cloud service also need to be provided by the target cloud service, otherwise some restructuring and re-architecting of the application is required. These factors influence the cost and effort and risk for porting the application.

It is possible that the resources available to applications in the target cloud service offer more functionality and thus can improve the application or reduce its size or complexity. Such factors could influence the cost and effort for porting the application.

9.3.2.2 Application syntactic portability for infrastructure capabilities type between CSPs

Application syntactic portability is similar to the non-cloud to cloud case. See 9.3.1.2.

9.3.2.3 Application instruction portability for infrastructure capabilities type between CSPs

Application instruction portability is similar to the non-cloud to cloud case. See 9.3.1.3.

9.3.2.4 Application metadata portability for infrastructure capabilities type between CSPs

Application metadata portability is similar to the non-cloud to cloud case. See 9.3.1.4.

9.3.2.5 Application behaviour portability for infrastructure capabilities type between CSPs

Application behaviour portability is similar to the non-cloud to cloud case. See 9.3.1.5.

9.3.2.6 Application policy portability for infrastructure capabilities type between CSPs

This type of portability is described in 9.6.

9.4 Application portability for platform capabilities type cloud services

9.4.1 Application portability from non-cloud to cloud service deployments

9.4.1.1 General

For this case, the application is being ported from a non-cloud environment deployment to a cloud service that offers platform capabilities.

The major characteristic of platform capabilities cloud services (such as a PaaS) is that they offer an environment in which to develop, deploy and operate applications. Such services typically involve the provision of diverse application software infrastructure functionality (also called "middleware"). This can include application runtimes, where the software stack required to run the application code is provided. It can also include the provision of substantial sets of services that can be used by the application code; reducing the amount of code that needs to be written, deployed and managed by the customer to achieve particular business objectives. Such services can cover almost any capability, but common examples include database systems, integration brokers, business process management systems, rules engines, analytics engines.

Porting an application to a platform capabilities cloud service can be done to take advantage of the managed runtime environment(s) offered, including the software stack required by the application. This is illustrated in Figure 15. The port may also benefit from the use of cloud services for some of the dependencies of the application, for example, a database. Even without modifying the application code in any way, it may be possible for the ported application to gain the benefits of the cloud environment, such as elasticity and scalability.

One of the common challenges in porting an application from a non-cloud deployment is defining the scope and boundaries of the application; identifying all the application artefacts and also identifying all the service dependencies of the application. Once all the artefacts and dependencies are identified, then it is necessary to define an approach for porting each of them.

It may be particularly straightforward to port an application to a platform capabilities cloud service where the non-cloud application is using a well-defined environment that is replicated in the cloud service. Examples include web servers and application servers, where the application code is relatively small and it depends on many capabilities which are supplied by the web server or the application server.

For application dependencies, it is possible to port the implementations of the dependencies to the platform cloud service. However, another strategy possible is to replace the non-cloud implementations of dependencies with components available in the cloud service environment, for example, replacing a customer installed and run on-premises database with a provider installed and run version of the same or an equivalent database.

Once the application is ported to the platform capabilities cloud service, there is a great opportunity to extend and enhance the application by connecting it to some of the additional services offered by the cloud provider. This ability to enhance the application rapidly may be one of the major business drivers behind the need to port the application. These services may offer significant business benefits such as support for big data analytics, mapping, "Internet of Things" support and machine learning and artificial intelligence capabilities. These services might be difficult to replicate in a non-cloud environment due to the specialist skills involved, but may be provided off-the-shelf in the cloud computing environment.

In some cases, the platform capabilities cloud service requires the use of some built-in capabilities that are supplied by the CSP and therefore, the equivalent capability in the original non-cloud solution needs to be replaced. An example of this is firewall capabilities, which are often built in to a platform capabilities cloud service.

A significant area of concern when migrating from a non-cloud deployment to deployment on a platform capabilities cloud service are development and test capabilities. It is common for platform capabilities cloud services to offer tools and services for development and test; these tools are usually different from those used in a non-cloud environment. It is necessary for the cloud service customer

to consider whether to port existing tools and services for development and test to the target cloud service environment or whether to adopt some or all of the tools and services offered by the target cloud service.

Table 12 describes considerations when porting cloud and non-cloud applications to services of cloud platform capabilities type. Expanded discussions of these considerations are provided in following sub-clauses.

Table 12 — Cloud application portability: Examples of consideration when porting both non-cloud and cloud applications and porting applications between clouds to cloud platform capabilities type

Cloud application portability facets	**Example considerations when porting applications to cloud platform capabilities type**	
	Non-cloud to cloud service	**Cloud service to cloud service**
Application syntactic	See 9.4.1.2.	See 9.4.2.2.
Application instruction	a) Definition and scope of application stack 1) Non-cloud applications may have a long and potentially less documented history within the organization. Establishing the boundaries of the application and its constituent parts may prove challenging. b) Control of application stack 1) Because the cloud service provides for many of the base services, the design and capabilities of these services will be fixed. This will likely require redesign of the parts of the application that utilizes such services. c) Application services d) Application supporting services e) Development and test capabilities 1) These capabilities are typically aligned to the target cloud service and can be significantly different from the equivalent capabilities in the source environment, requiring adaptation of the development and testing processes. f) Application instruction support 1) This may be a significant consideration depending on the amount of source code to be ported and the support for that particular instruction language or specification.	a) Definition and scope of application stack 1) This will vary considerably based on the originating capabilities type of the application. b) Control of application stack c) Application services 1) Where possible, existing cloud platform services need to be used, resulting in less code to port with potentially greater business and technical benefits. d) Application supporting services 1) Ensure that all necessary supporting services be ported to the cloud service. 2) The overall boundary of the application may need to be altered to include other supporting services in the port. Alternatively, they could be replaced with cloud services that are offered in the new environment. It may also be possible to leave these services in place and call them as necessary from the new environment. e) Development and testing services 1) These capabilities are typically aligned to the target cloud service and can be significantly different from the equivalent capabilities in the source environment, requiring adaptation of the development and testing processes. 2) Testing for infrastructure capabilities types is often based heavily around Virtual Machine instances which will not be the case for platform capabilities types. 3) The development process, including packaging and deployment, will be different in platform capabilities types. f) Application instruction support 1) Code translation and/or rewrite may be necessary if the same language and version is not supported in the new environment.

Table 12 *(continued)*

Application metadata	See 9.4.1.4.	See 9.4.2.4.
Application behaviour	a) Application input/output test cases b) Application technical test cases 1) Performance, security and other technical issues may be benchmarked and compared. c) Business process flows 1) Additional business enhancements provided by the cloud platform (along with other reasons for the port) may be measured. 2) Because the port to a cloud platform capabilities type is often driven by business and not technical concerns, it is very likely that changes to the overall business processes surrounding the application are desired by the business. Therefore, such an application port may involve a significant change management project as well.	
Application policy	See 9.6.	

9.4.1.2 Application syntactic portability

Application syntactic portability means that the target cloud service needs to understand the format of an application and its related components as transmitted by the CSC. It is commonly the case that cloud services offering platform capabilities offer environments that support specific application frameworks and languages and that support for the framework and language already being used on-premises by the customer application is a major factor in the choice of the cloud service.

For example, the customer application might be a Node.js web application; it is typically the case that the target cloud service supports this language and so accepts the application artefacts, such as JavaScript files, directly with no changes.

9.4.1.3 Application instruction portability

Application instruction portability depends on the nature of the application artefacts and the target runtime environment. For example, applications built using interpreted languages such as Node.js or Python normally port to any target system containing the respective runtime environment. Languages compiled to machine instructions, such as C, need to be compiled to a particular target environment. Recompilation might be necessary if the target cloud service has an environment which does not match that of the non-cloud enterprise system, e.g. different processor type on the target system.

Lack of support for the required runtime environment for the application artefacts in the target cloud service implies a substantial porting cost usually requiring rewriting of the application, which is often so high that it means that the port to such a target environment is not attempted. However, smaller differences also need consideration, such as the target cloud service supporting a different version of the runtime than used in the source environment. This might require some adjustments to the application artefacts and retesting against the target cloud service.

Beyond the instructions of the application itself, there are various dependencies which the application can have. These dependencies can exist in the runtime software stack used by the application or they can exist as separate components or services used by the application. In either case, the dependencies need to be satisfied in the target platform cloud service, either through the runtime stack provided or through services available as part of the cloud service or associated with it or if necessary, services supplied by the customer. It is necessary for the dependencies to match those in the non-cloud environment as far as their APIs or invocation interfaces or else the code of the application has to be adapted to match those changed interfaces.

9.4.1.4 Application metadata portability

The application metadata includes information about resource requirements, deployment and configuration information and initialization data. It might be possible to port the application metadata unchanged between the non-cloud system and the platform cloud service, but such direct portability is uncommon. It is likely that the application metadata undergoes some level of transformation to match the format and content required by the platform cloud service.

9.4.1.5 Application behaviour portability

Obtaining application behaviour portability can be a significant challenge where the application runtime, software stack and service dependencies are different between the non-cloud environment and the platform cloud service.

In some cases, there may be version differences between directly equivalent software components on the two systems and these can introduce behaviour changes. In other cases, different software components may be used to satisfy particular dependencies. There may also be behavioural differences introduced by the nature of the platform cloud service itself, where for example, there may be limits on memory available or on the amount of processor power available to a single instance of the application.

9.4.1.6 Application policy portability

This is described in 9.6.

9.4.2 Application portability from one cloud service to another cloud service

9.4.2.1 General

In this scenario, the application is ported from one cloud service to a target cloud service where both services are platform capabilities type. Note that the actual CSP might be the same in both cases, but that it is typically not the case.

The existing application deployment can vary in a number of ways and therefore, the portability requirements also vary in consequence.

A typical example is to port a three-tier web application from running on one platform cloud service to a target platform cloud service of a different CSP. In such a scenario, an organization moves a three-tier application from one cloud platform provider to another. Actors involved include CSPs, software developers and system administrators/operators.

As shown in Figure 15, the application artefacts need to be ported in a way that enables them to function in the target cloud service. For example, the instruction sets and the configuration data will need to be understood and processed by the resources available on the target cloud service. The programming language used for the instruction sets need to be supported by the runtime in the target cloud service. Other examples of resources that need to be available are developer tools, runtimes, storage and database systems. Operating system APIs used by the application in the source cloud service are necessary in the target cloud service. Otherwise, developers have to modify the source code to utilize the APIs available in the target cloud service. This could affect the cost, effort and risk for the port.

In general, the target CSP needs to support the platform services that the application uses in the source cloud service. What platform services the application relies on varies and depends on the application's architecture and the choices made in the design of the application to rely on the platform capabilities offered by the source cloud service. Such capabilities are marked in Figure 15 in components that have dashed borders. The process of porting the application to a target cloud service involves considering the dependencies and how they are satisfied. In some cases, the platform dependencies are available in an identical manner on the target cloud service and in other cases, adaptation of the application to the platform capabilities available on the target cloud service is necessary. The extent of the changes influences the cost, effort and risk involved in the port.

The cost and effort of porting a cloud application is decreased if the application's service dependencies, i.e. the components shown in dashed boxes in Figure 15, are abstracted to shield the application from the specifics of a given CSP implementation.

This is especially relevant given many CSPs offer platform capabilities type of cloud services that are similar and yet accessing those services is accomplished through different REST APIs and SDKs. Abstracting out the application's internal logic that accesses the dashed boxes in Figure 15 would ease the mapping of functionality offered by such SDKs. This requires that there be a separation between application logic and the underlying cloud platform services (shown as dashed boxes in Figure 8 and Figure 9). Ideally, when possible, the application logic should be agnostic of which cloud platform it is running on, using abstraction layers.

Common examples of prescriptive cloud application patterns and practices for such abstraction are:

— blob, relational or NoSQL storage;

— cache management;

— load balancing;

— inter-component messaging queues;

— identity and access management.

Another case of portability is when the application is invoking services offered via CSP:inter-cloud mechanisms of intermediation, aggregation, federation or arbitrage (see ISO/IEC 17789). Portability of such applications is subject to availability of the CSP:inter-cloud services, when the application is ported to the target cloud service, otherwise, the application porting cost and effort and risk increase since adapting to use other services to meet its dependencies is necessary.

Development and test capabilities are necessary considerations when porting an application between two different cloud services. It may be the case that the tools and services for development and test are different between the source cloud service and the target cloud service. If this is the case, then it is necessary to adapt the development and test capabilities when porting the application.

9.4.2.2 Application syntactic portability

It is desirable for the target cloud service to accept the format of the application artefacts and that of related components, as deployed in the source cloud service. If this is not possible, then some amount of conversion or adaptation is necessary which increases the cost, effort and risk of porting the application. See also 9.4.1.2.

9.4.2.3 Application instruction portability

See 9.4.1.3.

9.4.2.4 Application metadata portability

See 9.4.1.4.

9.4.2.5 Application behaviour portability

See 9.4.1.5.

9.4.2.6 Application policy portability

This type of portability is described in 9.6.

9.5 Application portability for application capabilities type cloud service

In the case of application capabilities type cloud service, the cloud service typically offers a complete application of some kind. The application code for the cloud service is installed and operated by the CSP; the code does not come from the CSC. For this reason, there is no porting of an application to or from a cloud service of this type.

Where a cloud service customer has an existing non-cloud application, moving to use an application capabilities type cloud service with equivalent capabilities is typically a replacement of the non-cloud application. The same is true when moving between two equivalent application type cloud services. In both cases, no code artefacts are ported.

Application capabilities type of services do not offer the necessary capabilities for porting applications. Such services do not offer infrastructure or platform capabilities to support applications, so there is no provision for porting application code.

The only meaningful portability type for the case of application capabilities cloud services is data portability, which is addressed separately in Clause 8.

9.6 Application portability: Policy facet

9.6.1 General

This document does not attempt to describe all of the possible legal and organizational implications around application portability. This document describes the general types of issues that may be encountered in the real world, so that these can be considered and addressed with appropriate and qualified legal advice.

9.6.2 Law and regulations

Application portability scenarios might be restricted by specific matters of statute law (or civil code, common law, legal precedent or other form of law), or by regulations set and enforced by authorized government departments or agencies. This is most likely for particular vertical industries such as government, defense, security, healthcare, financial, child welfare, education, charitable organizations, etc.

These laws and regulations might impose specific requirements that might complicate or prevent the movement of applications (and their associated data) from customers to and from providers and between providers. For example, there might be limitations on the geographic location of such applications (where the data is to be processed) or where the data is to be stored. There might be specific criteria that service providers have to meet, such as security clearances, compliance with specific standards, export controls or prior approval by a government agency.

A further consideration arises for some types of applications such as games or video content which may not be acceptable in some countries. The location of servers for such games or content could then become an issue.

Some territories can also have laws or regulations covering the monitoring, recording, retention, filtering or tracking of user activity in some types of applications running within their jurisdiction. Thus, the choice of a new CSP and the location of their data centres, can affect the privacy of customers, which could require changes to customer terms and conditions or renegotiation of contracts. This can also affect the actual process of porting the application, since it might be necessary to “screen” application data as it migrates from one jurisdiction to another.

Some territories also have strict laws and regulations around correct use of character sets in user interfaces (notably Asian characters and their usage), legal requirements to support mandatory “official” languages (some of which might have few real speakers) and regulations governing accessibility of the application to those with disabilities or other challenges.

It is sometimes possible to avoid these issues if the application is based elsewhere, but perhaps harder to justify when it is served from a data centre within the regulating territory. While it is to be hoped that some harmonization of these various legal and regulatory regimes will evolve over time to make this process easier, at least at a regional level, the current situation does not allow for this.

9.6.3 Contracts and licenses

In addition to public law and regulations as mentioned above, software applications (and some forms of licensed data such as maps, media, demographic data sets, etc.) are also governed by private law in the form of signed contracts and licenses granted by the originator of the content. Some of these agreements and licenses can pre-date the advent of cloud computing, which can give rise to ambiguity or disputes if the application is moved to a cloud service.

Applications, unless written in-house or written under contract for the customer, are often licensed in whole or in part from one or more software vendors. If the original license(s) for the application (or components thereof) is written assuming a non-cloud deployment on traditional client or server hardware, it might contain language that is not directly applicable to cloud deployment. For example, it might say that the application can be deployed "on a single computer" without any mention of virtual machines, hosting, cloud computing or other concepts. The porting of such software to a cloud service might require permission from the original vendor and the payment of additional fees.

Some applications might be designed to work specifically with a single cloud service. The vendor might have focussed on developing their application for a specific cloud service vendor as part of an exclusivity agreement and be unwilling or unable to relicense the software for use elsewhere. It might be that any warranties over the reliability of the application are based on it running in an environment that the vendor has tested and approved. Porting the application to another cloud service might invalidate such warranties.

Some applications might be licensed for use on servers in specific geographical territories. Among other reasons, this could arise because the software vendor wishes to protect their in-country salesforce from being bypassed by customers placing orders for software in other territories where market prices or currency exchange are more advantageous. The migration of such a license for a cloud service potentially operating across multiple territories is likely to require renegotiation with the vendor.

9.6.4 Organizational policies

An organization can have their own internal policies that directly affect application portability.

The most obvious of these are organization decisions that require "qualification" of vendors. This can be as simple as having the necessary financial paperwork in place to be on the "approved vendor" list for the organization. It can be more complex (especially for government departments) involving extensive rules around compliance with certain standards, certifications, security clearances of staff or other criteria. In some cases, these rules might predate cloud computing which can make them hard to apply for a cloud environment. For example, if some vendor acceptance criteria require allowing government inspectors to have physical access to the vendor's servers, it probably will not work in a public cloud environment, where such servers are virtual and likely move around, sharing hardware with other customers.

Other organizational policies can also affect porting to specific CSPs. For example, a CSC might have a policy of only using service providers with a good "green" record of environmental responsibility or of only using services based in countries which meet certain ethical standards decided by the organization or a preference for CSP companies that are minority-led, female-led, etc. There may also be operational policies such as requirements for service diversity, geo-redundancy, accessibility, privacy, mandatory support of minority user languages, etc., beyond those required by law.

Bibliography

[1] ISO 15836, *Information and documentation — The Dublin Core metadata element set*

[2] ISO/TR 20943-5, *Information technology — procedures for achieving metadata registry content consistency — Part 5: Metadata mapping procedure*

[3] ISO/IEC 17203, *Information technology — Open Virtualization Format (OVF) specification*

[4] ISO/IEC 17788, *Information technology — Cloud computing — Overview and vocabulary*

[5] ISO/IEC 17789, *Information technology — Cloud computing — Reference architecture*

[6] ISO/IEC 19086-1, *Information technology — Cloud computing — Service level agreement (SLA) framework — Part 1: Overview and concepts*

[7] ISO/IEC 19944:2017, *Information technology — Cloud computing — Data and their flow across devices and cloud services*

[8] ISO/IEC 21320-1, *Information technology — Document Container File — Part 1: Core*

[9] ISO/IEC 27000, *Information technology — Security techniques— Information security management systems— Overview and vocabulary*

[10] ISO/IEC 27001, *Information technology — Security techniques— Information security management systems— Requirements*

[11] ISO/IEC 27017, *Information technology — Security techniques— Code of practice for information security controls based on ISO/IEC 27002 for cloud services*

[12] ISO/IEC 27018, *Information technology — Security techniques— Code of practice for protection of personally identifiable information (PII) in public clouds acting as PII processors*

[13] European Interoperability Framework (EIF) Towards Interoperability for European Public Services. Available from <http://ec.europa.eu/isa/documents/eif_brochure_2011.pdf>

[14] Wang W.G., Tolk A., Wang W.P. (2009), *The Levels of Conceptual Interoperability Model: Applying Systems Engineering Principles to M&S*, Available from <http://arxiv.org/ftp/arxiv/papers/0908/0908.0191.pdf>

[15] Cloud Standards Customer Council. (2014), *Practical Guide to Cloud Computing V2.0*, Available from <http://www.cloud-council.org/deliverables/CSCC-Practical-Guide-to-Cloud-Computing.pdf>

[16] Regulation (EU). 2016/679, *General Data Protection Regulation*, Available from <http://ec.europa.eu/justice/data-protection/reform/files/regulation_oj_en.pdf>

[17] NIST FIPS. 140-2 (including change notices as of 12-03-2002) Security Requirements for Cryptographic Modules, Available from <http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-2.pdf>

[18] IETF RFC. 4511 (2006): *Lightweight Directory Access Protocol (LDAP)*, Available from <https://tools.ietf.org/html/rfc4511>

[19] IETF RFC. 6749 (2012): *The OAuth 2.0 Authorization Framework*, Available from <https://tools.ietf.org/html/rfc6749>

[20] Open I.D. (2014): *OpenID Connect*, Available from <http://openid.net/connect/>

[21] OASIS. (2005): *Security Assertion Markup Language (SAML) 2.0*, Available from <http://saml.xml.org/saml-specifications>

[22] PCI Security Standards Council. (2011), *PCI Data Security Standard (PCI DSS)*, Available from <https://www.pcisecuritystandards.org/documents/Virtualization_InfoSupp_v2.pdf>

ICS 35.020; 01.040.35

Price based on 65 pages