Document Identifier: DSP-IS0501

Date: 2015-10-04

Version: 1.0.0m

Software Defined Data Center (SDDC) Definition
A White Paper from the OSDDC Incubator

Information for Work-in-Progress version:

IMPORTANT: This document is not a standard. It does not necessarily reflect the views of the DMTF or all of its members. Because this document is a Work in Progress, it may still change, perhaps profoundly. This document is available for public review and comment.

Provide any comments through the DMTF Feedback Portal:
http://www.dmtf.org/standards/feedback

Supersedes: None

Document Class: Informative

Document Status: Work in Progress

Document Language: en-US

CONTENTS

Figures

Tables

Foreword

The *Software Defined Data Center (SDDC) Definition* (DSP-IS0501) was prepared by the Open Software Defined Data Center (OSDDC) Incubator.

The goal of the OSDDC Incubator is to develop SDDC use cases, reference architectures, and requirements based on real world customer requirements. Based on these inputs, the Incubator will develop a set of white papers and set of recommendations for industry standardization for the SDDC.

The work coming out of this incubator will result in:

1. Clear definition and scope of the SDDC concept.
2. New work items to existing chartered working groups.
3. Expanded scope to existing chartered groups
4. Creation of new working groups, if needed.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. For information about the DMTF, see http://www.dmtf.org.

Acknowledgments

The DMTF acknowledges the following individuals for their contributions to this document:

- Ali, Ghazanfar - ZTE Corporation
- Black, David - EMC
- Bumpus, Winston - VMWare, Inc.
- Carlson, Mark - DMTF Fellow
- Dolin, Rob - Microsoft Corporation
- Khasnabish, Bhumip - ZTE
- Leung, John - Intel
- McDonald, Alex - NetApp
- Ronco, Enrico - Telecom Italia
- Snelling, David - Fujitsu
- Shah, Hemal - Broadcom
- Wells, Eric - Hitachi, Ltd.
- Wheeler, Jeff - Huawei
- Zhdankin, Alex - Cisco

Software Defined Data Center (SDDC) Definition

1 Executive summary

1.1 Introduction

The Software Defined Data Center (SDDC) is an evolutionary result of virtualization and cloud computing technologies. To date, the SDDC has been defined in many ways. The following examples are a few of the more prevalent (and realistic) definitions gleaned from a large number of resources used for this paper:

> *"A Software Defined Data Center (SDDC) is a data storage facility in which all elements of the infrastructure – networking, storage, CPU and security – are virtualized and delivered as a service. Deployment, provisioning, configuration and the operation, monitoring and automation of the entire infrastructure is abstracted from hardware and implemented in software." (Forrester)*

Another:

> *"SDDC is the phrase used to refer to a data center where the entire infrastructure is virtualized and delivered as a service." (VMware)*

It is clear that the move to the SDDC is a major technology shift. While other definitions have been proposed by various vendors, they all have similar intent.

The goal of this paper is to outline use cases, and definitions, and identify existing standards gaps, and possible architectures for the various implementations of SDDC.

1.2 SDDC definition

Software Defined Data Center (SDDC): a programmatic abstraction of logical compute, network, storage, and other resources, represented as software. These resources are dynamically discovered, provisioned, and configured based on workload requirements. Thus, the SDDC enables policy-driven orchestration of workloads, as well as measurement and management of resources consumed.

The SDDC comprises a set of features that include:

- Logical compute, network, storage, and other resources
- Discovery of resource capabilities
- Automated provisioning of logical resources based on workload requirements
- Measurement and management of resources consumed
- Policy-driven orchestration of resources to meet service requirements of the workloads

2 Use cases

This clause describes use cases for various services that can be provided by an SDDC, including Infrastructure as a Service (IaaS) and Software as a Service (SaaS).

2.1 Infrastructure as a Service (IaaS)

In IaaS, the customer wants to execute a workload and uses the data center to host the infrastructure. After the infrastructure is available, the customer installs the necessary software and content/data, then executes the workload.

Figure 1 shows the interactions in an IaaS environment based on a software-defined data center.

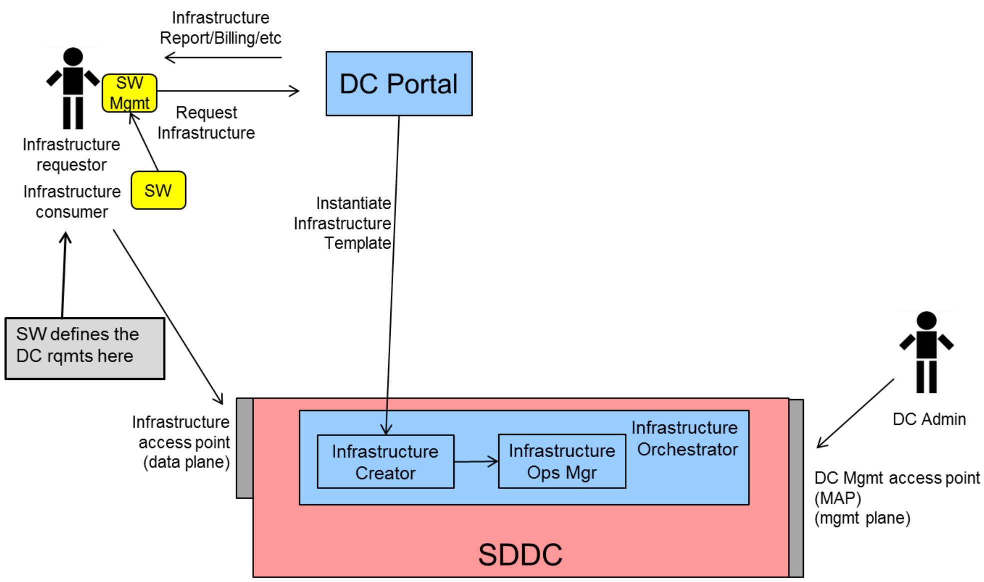

Figure 1 - IaaS use case for SDDC

2.1.1 Actors

There are two actors: the customer and the IaaS data center (DC) administrator. The customer has two aspects: the infrastructure requestor and the infrastructure consumer.

The infrastructure requestor performs the following tasks:

- Designs an application composed of a workload that executes on a specific compute/storage topology
- Requests an infrastructure with specific workload requirements
- Verifies infrastructure (including firmware/BIOS)
- Requests that infrastructure be increased or decreased
- Receives usage reports and billing

The infrastructure consumer performs the following tasks:

- Installs the OS, and applications and delivers content
- Starts the workload

The IaaS DC administrator performs the following tasks:

- Monitors power and cooling in the data center
- Adds (or replaces) platforms/resources to the data center
- Receives notification of resource depletion (or surplus?)
- Takes inventory (accounting, SW licenses, etc.)
- Performs security audit (or sec. contractor)

- Receives notification of potential brown-outs
- Updates platform firmware (security, etc.)

2.1.2 Use Case

The workload is known to the infrastructure requestor.

In the diagram, the flow proceeds as follows:

1. The infrastructure requestor inspects the workload (WL) and determines the infrastructure to request.
2. The infrastructure requestor requests an infrastructure with specific service requirements from the service portal.
3. The service portal makes a request to the infrastructure orchestrator to instantiate the infrastructure.
4. The infrastructure orchestrator instantiates the infrastructure.
5. The infrastructure starts the infrastructure. At this point, both the infrastructure move to the operational phase and are managed by the infrastructure operation manager.
6. Once running, the infrastructure is available to the infrastructure consumer.

2.2 Software as a Service (SaaS)

In SaaS, the customer wants to instantiate a service and uses the data center to host the service. The service may be consumed by a service consumer, which is distinct from the SaaS customer. Once the service is instantiated the service requestor may need to provide additional content before the service is enabled and ready to be consumed.

Figure 2 shows the interactions in a SaaS environment based on a software-defined data center.

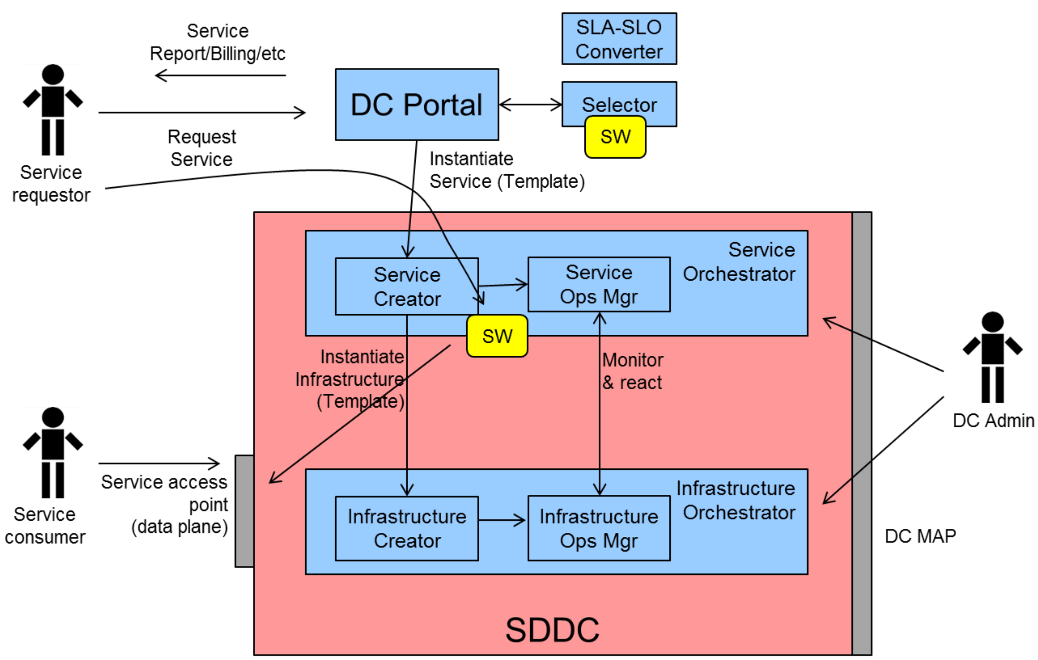

Figure 2 - SaaS use case for SDDC

2.2.1 Actors

There are three actors: the service requestor, the service consumer, and the SaaS DC administrator.

The service requestor wants to instantiate a service and performs the follow tasks:

- Requests a service with specific service requirements
- Monitors the service
- Changes the service requirements of an operational service
- Requests that the service scales up or scales down
- Requests migration of the service to another service provider
- Requests the service be terminated

The service consumer performs the following task:

- Uses the service

The SaaS DC administrator performs the following tasks:

- Monitors the service
- Monitors power and cooling in the data center
- Adds (or replaces) platforms/resources in the data center
- Receives notification of resource depletion (or surplus?)
- Takes inventory (accounting, SW licenses, etc.)
- Performs security audits (or sec. contractor)
- Receives notification of potential brown-outs
- Stages/tests new services
- Updates platform firmware (security, etc.)

2.2.2 Use case

The workload that defines the service infrastructure is known to the DC service portal. In the diagram, the flow proceeds as follows:

1. The service requestor requests a service with specific service requirements from the service portal.
2. If multiple service templates are possible, the service portal or the service requestor may select the specific service template.
3. The service portal makes a request to the service orchestrator to instantiate the service.
4. The service creator makes a request to the infrastructure orchestrator to instantiate the infrastructure.
5. After the infrastructure is instantiated, the service creator installs the OS, applications, and the content and configures accordingly.
6. Finally, the service creator starts the service and the service is available to the server consumer
7. At this point, both the infrastructure and service move to the operational phase and are managed by their respective operation managers.

3 SDDC technology and functionality

An SDDC incorporates and is heavily dependent upon the use of topologies that abstract, optionally pool, and automate the use of the virtualized resources. Virtualization technologies can be thought of as common resources when integrated and used by the SDDC. The focus on industry standardized management models and application programming interfaces (APIs) provide this level of abstraction. Various vendors and SDOs are championing their respective offerings into the new SDDC community.

The SDDC comprises a set of features that include:

1. Logical compute, network, storage and other resources

2. Discovery of resource capabilities
3. Automated provisioning of logical resources based on workload requirements
4. Measurement and management of resources consumed
5. Policy-driven orchestration of resources to meet service requirements of the workloads

Additional SDDC features and functionalities include:

- Topology automation
- Security (authentication, authorization, auditing), intrusion detection system (IDS), intrusion prevention system (IPS), firewall

The SDDC should be:

- Standardized – API and functional model
- Holistic – system wide abstractions
- Adaptive - elasticity driven by the workload
- Automated - provisioning, configuration, and run-time management

3.1 SDDC, virtualization and cloud relationships

Virtualization is central to the SDDC but in itself is not sufficient. The three major building blocks that virtualization delivers are: compute, storage, and network:

1. Compute Virtualization – Abstraction of compute resources that can be realized with underlying collection of physical server resources. This concept includes abstraction of the number, type, and identity of physical servers, processors, and memory. Other technologies, such as containers, may also be used.
2. Storage Virtualization – Abstraction of storage resources that can be realized with underlying physical and logical storage resources. This concept includes abstraction of the number, type, and identity of physical disks.
3. Network Virtualization - Abstraction of network resources that can be realized using underlying physical and logical resources. This concept includes abstraction of the number, type, and identity of physical media, connectivity, and protocol.

4 SDDC architectures

Building on virtualization technology through standard APIs allows the SDDC automation to provision exactly those resources required for the software that will be deployed on those resources. This is shown in the lowest two layers of Figure 3 as the Data center Abstraction Layer (DAL) and Virtualization and Resource Characterization layer. This automation is envisioned to interpret the requirements for the deployed software and configure the resources appropriately to meet those requirements. The requirements may be conveyed to the administrator out of band, as is typical today, and in this case the administrator must interpret these requirements. However the requirements may also be conveyed through an API, the implementation of which interprets the requirements and automates what the administrator would otherwise need to do manually. This is shown in Figure 3 with the thin black arrows being the manual requirements conveyed to the administrator and the results of the administrators interpretation conveyed manually, out of band, back as service levels. The administrator responds by providing resources that will meet the service levels required by the software. The blue arrow represents a self-service management interface that incorporates elements with the ability to convey the Compute, Storage and Networking requirements in-band such that the manual, out-of-band, requirements path is no longer needed. This has been identified as a gap for such interfaces as DMTF CIMI. The requirements need to be abstracted and added to the interface as metadata for the various loads that need resources.

Short term, the Infrastructure Service Characteristics shown in the top box as Provisioning, Protection, Availability, Performance, Security, and Energy Consumption are typically implemented for coarse-grained virtual and in some cases physical resources. Thus while the resources themselves may be virtualized and provisioned with fine-grained control (provisioned at the granularity of individual workloads), the services that provide these characteristics may not. To accommodate this, the top box contains pools of resources configured and provisioned at this coarse granularity with the coarse-grained services. Resource pooling is a technique used for various reasons and includes similarly configured resources both unused and already provisioned. We use some example pool names for clarity, but there may be many differently configured pools from which to draw. This way the administrator, if he is manually interpreting the requirements, can simply pick the pool with the best match of resource configurations for those requirements. If there is similar automation software receiving the requirements via the self-service interface, that software can do the interpretation and select the correct pool with an algorithm. We see this resource pooling technique as a temporary approach that should be obviated after the infrastructure services are able to act at a finer grained level.

SDDC builds upon virtualization technology by expanding the scope from individually virtualized components to the entire data center, and envisions a unified control and management solution.

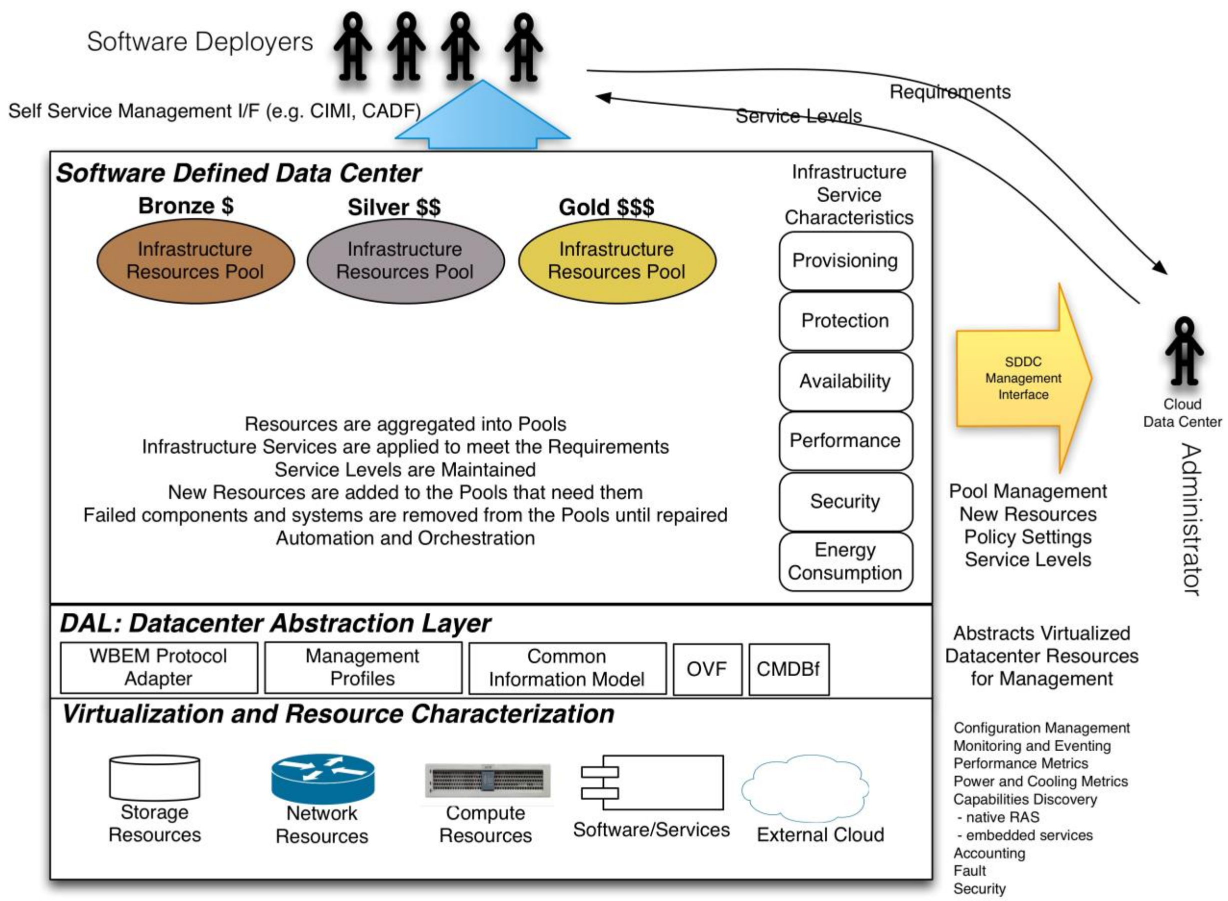

Figure 3 - SDDC architecture

Figure 3 shows all the elements of an SDDC. The SDDC architecture defines data center resources that include software-based services. The DAL layer provides abstraction of compute, network, and storage resources, which are then virtualized and configured according to the requirements of the workload.

The DAL is a unifying and consistent abstraction for the underlying resources and provides a standardized interface and common model that may be used by the SDDC management automation software.

4.1 Server virtualization

Server virtualization releases CPU and memory from the limitations of the underlying physical hardware. As a standard infrastructure technology, server virtualization is the basis of the SDDC, which extends the same principles to all infrastructure services.

4.2 Software Defined Network

In a Software Defined Network (SDN), the network control plane is moved from the switch to the software running on a server. This improves programmability, efficiency, and extensibility. SDN is to date the most developed and understood software-defined technology. Therefore this paper does not delve into the details of this software-defined component.

4.3 Software Defined Storage

Software Defined Storage (SDS) is an emerging ecosystem of products and requires further discussion here. This software should make visible all physical and virtual resources and enables programmability and automated provisioning based on consumption or need. SDS separates the control plane from the data plane and dynamically leverages heterogeneity of storage to respond to changing workload demands. The SDS enables the publishing of storage service catalogs and enables resources to be provisioned on-demand and consumed according to policy.

In many respects, SDS is more about packaging and how IT users think about and design data centers. Storage has been largely software defined for more than a decade: the vast majority of storage features have been designed and delivered as software components within a specific, storage-optimized environment.

The Storage Networking Industry Association (SNIA) definition of SDS allows for both proprietary and heterogeneous platforms. To satisfy the SNIA definition, the platform must offer a self-service interface for provisioning and managing virtual instances of itself.

4.3.1 Necessary SDS functionality

Because many storage offerings today have already been abstracted and virtualized, what capabilities should be offered to claim the title of Software Defined Storage?

Software Defined Storage should include:

- **Automation** – Simplified management that reduces the cost of maintaining the storage infrastructure.
- **Standard Interfaces** – APIs for the management, provisioning, and maintenance of storage devices and services.
- **Virtualized Data Path** – Block, File, and Object interfaces that support applications written to these interfaces.
- **Scalability** – Seamless ability to scale the storage infrastructure without disruption to availability or performance.

Ideally, SDS offerings allow applications and data producers to manage the treatment of their data by the storage infrastructure without the need for intervention from storage administrators, without explicit provisioning operations, and with automatic service level management. In addition, data services should be able to be deployed dynamically and policies should be used to maintain service levels and match the requirements with capabilities. Metadata should be used to:

- Express requirements
- Control the data services
- Express service level capabilities

4.4 Data center Abstraction Layer

The Data center Abstraction Layer (DAL) is a unifying and consistent abstraction for the virtual and physical resources within the data center. It extends the concept of a Hardware Abstraction Layer (HAL) to the entire data center.

Prior to the development of the HAL, operating systems and applications were dependent on specific features provided by the hardware of the PC architecture. By adopting standard protocols, the HAL provided an abstract interface that allowed these variations to be isolated from the operating systems and applications.

In a similar manner the DAL abstracts variations in data center compute, network, storage, and software resources, presenting them as standardized resources within the SDDC.

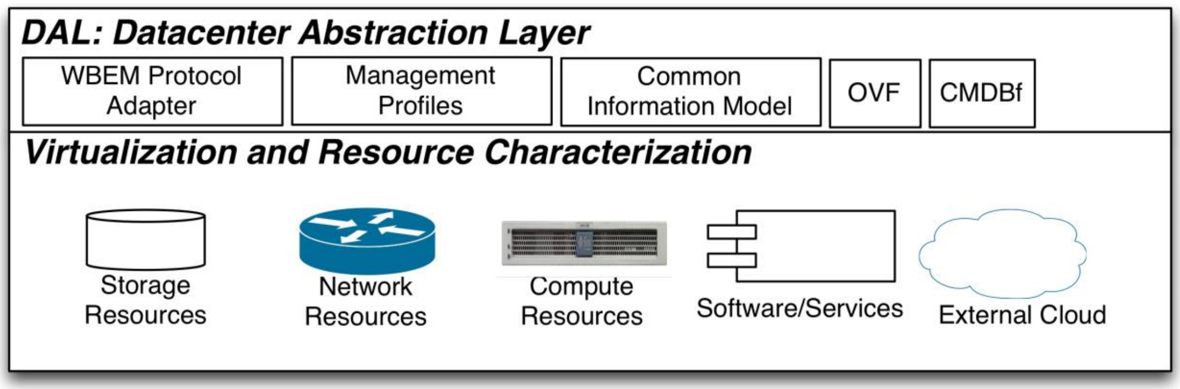

Figure 4 - Data center Abstraction Layer

The DAL enables:

- Management layers in the SDDC to manage resources in a consistent manner
- Introduction of new resources without requiring changes to the management or application layers

Improved efficiency and utilization of resources by the SDDC

4.5 Trust Boundary and Multi-Tenant Isolation Requirements

As shown in Figure 3(SDDC Architecture), it is expected that in a typical SDDC implementation, virtualized computing, networking, storage and other resources will be shared by multiple tenants who are hosted in the same set of physical devices.

It is therefore imperative that explicit trust boundaries are set among these tenants in order to maintain appropriate isolation among the often competing tenants. Without proper isolation, policy, security, and automation related information may be compromised and these in turn may result in loss of revenue for the well-behaved tenants.

From the applications, services and administration viewpoints, it may be required to support tenancy-specific resources and their configurations including service-quality (resiliency), even when the needs span multiple physical devices in multiple physical locations. This may need to be achieved by using tenancy-specific embedded authorization and authentication.

Trust boundary can be established using perimeters for physical, logical, address space, domain and topology segmentation, peering and routing profiles, and so on.

It may be required to routinely monitor and log tenant's identification, credentials, service and resources-usage contracts, etc. so that these can be frequently verified and updated in order to prevent spoofing or other types of attacks.

5 Applicable standards activity

While the DMTF is currently the only SDO specifically focusing on developing models for the SDDC, many other organizations have work that is relevant. Work in other SDOs is mainly focused on SDN and SDS, but it is important to look at emerging standards and how they may be relevant to SDDC.

5.1 DMTF

DMTF standards enable effective management of IT environments through well-defined interfaces that collectively deliver complete management capabilities. DMTF standard interfaces are critical to enabling interoperability among multivendor IT infrastructures, and systems and network management including cloud computing, virtualization, desktop, network, servers, and storage.

Some of the key DMTF standards and initiatives under development that will enable the new SDDC paradigm are described below.

Open SDDC Incubator

The DMTF is the only SDO currently that is focusing on developing initial management models for the SDDC marketplace. The DMTF recently launched its 'SDDC Incubator' with the charter of directing all future work in the DMTF for SDDC.

Cloud Management Initiative

The DMTF's Cloud Management Initiative is focused to promote interoperable cloud infrastructure management between cloud service providers and their consumers and developers. Working groups within the initiative develop open standards with the aim of achieving this interoperability.

Network Management Initiative

DMTF's Network Management Initiative (NETMAN) is an integrated set of standards for management of physical, virtual, application-centric, and software-defined networks. The NETMAN initiative aims at unifying network management across traditional data centers, cloud infrastructures, NFV environments, and SDDC ecosystems.

Virtualization Management Initiative

DMTF's Virtualization Management (VMAN) initiative includes a set of specifications and profiles that address the management life cycle of a heterogeneous virtualized environment.

5.1.1 Cloud Infrastructure Management Interface (CIMI)

CIMI is a high-level, self-service, interface for infrastructure clouds that greatly simplifies cloud systems management, allowing users to dynamically provision, configure, and administer their cloud usage. The specification standardizes interactions between cloud environments, using JSON and XML, to achieve interoperable cloud infrastructure management.

CIMI was adopted as an International Standard by the Joint Technical Committee 1 (JTC 1) of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) in March 2015.

Version 2 of the CIMI specification, which is currently under development, extends the previous work with an enhanced network model and modelling of multicloud and intercloud scenarios.

5.1.2 Open Virtualization Format (OVF)

The OVF specification provides a standard format for packaging and describing virtual machines and applications for deployment across heterogeneous virtualization platforms. OVF was adopted by the American National Standards Institute (ANSI) in August 2010 and as an International Standard in August 2011 by the Joint Technical Committee 1 (JTC 1) of the International Organization for Standardization (ISO), and the International Electrotechnical Commission (IEC). In January 2013, DMTF released the second version of the standard, OVF 2.0, which applies to emerging cloud use cases and provides important developments from OVF 1.0 including improved network configuration support and package encryption capabilities for safe delivery.

5.1.3 Web-Based Enterprise Management (WBEM)

Web-Based Enterprise Management (WBEM) is a set of specifications that define how resources can be discovered, accessed, and manipulated, facilitating the exchange of data across otherwise disparate technologies and platforms.

WBEM defines protocols for the interaction between systems management infrastructure components implementing the Common Information Model (CIM), and is a major component of the DAL.

5.1.4 Common Information Model (CIM)

The CIM Schema is a conceptual schema that defines how managed elements in an IT environment are represented as a common set of objects and relationships. CIM is extensible in order to allow product specific extensions to the common definition of these managed elements. CIM uses a model based upon UML to define the CIM Schema and is the basis for most other DMTF standards.

5.1.5 Configuration Management Database Federation (CMDBf)

CMDBf facilitates the sharing of information between configuration management databases (CMDBs) and other management data repositories (MDRs). The CMDBf standard enables organizations to federate and access information from complex, multivendor infrastructures, simplifying the process of managing related configuration data stored in multiple CMDBs and MDRs.

5.1.6 Systems Management Architecture for Server Hardware (SMASH)

DMTF's SMASH standards are a suite of specifications that deliver architectural semantics, industry standard protocols and profiles to unify the management of the data center. The SMASH Server Management (SM) Command Line Protocol (CLP) specification enables simple and intuitive management of heterogeneous servers in the data center. SMASH takes full advantage of the DMTF's Web Services for Management (WS-Management) specification - delivering standards-based Web services management for server environments. Both provide server management independent of machine state, operating system state, server system topology, or access method, facilitating local and remote management of server hardware. SMASH also includes the SM Managed Element Addressing Specification, SM CLP-to-CIM Mapping Specification, SM CLP Discovery Specification, SM Profiles, as well as a SM CLP Architecture White Paper.

5.1.7 Redfish API

Scalability in today's data center is increasingly achieved with horizontal, scale-out solutions, which often include large numbers of simple servers. The usage model of scale-out hardware is drastically different from that of traditional enterprise platforms, and requires a new approach to management.

The DMTF's Redfish API is an open industry standard specification and schema designed to meet the expectations of end users for simple, modern, and secure management of scalable platform hardware. The Redfish API specifies a RESTful interface and utilizes JSON and OData to help customers integrate solutions within their existing tool chains.

5.2 OASIS

OASIS (Organization for the Advancement of Structured Information Standards) is a nonprofit, international consortium whose goal is to promote the adoption of product-independent standards for information formats.

5.2.1 Cloud Application Management for Platforms (CAMP)

OASIS CAMP advances an interoperable protocol that cloud implementers can use to package and deploy their applications. CAMP defines interfaces for self-service provisioning, monitoring, and control. Based on REST, CAMP is expected to foster an ecosystem of common tools, plug-ins, libraries, and frameworks, which will allow vendors to offer greater value-add.

Common CAMP use cases include:

- Moving on-premises applications to the cloud (private or public)
- Redeploying applications across cloud platforms from multiple vendors

5.2.2 Topology and Orchestration Specification for Cloud Applications (TOSCA)

The TOSCA TC substantially enhances the portability of cloud applications and the IT services that comprise them running on complex software and hardware infrastructure. The IT application and service level of abstraction in TOSCA will also provide essential support to the continued evolution of cloud computing. For example, TOSCA would enable essential application and service life cycle management support, e.g., deployment, scaling, patching, etc., in Software Defined Environments (SDE), such as Software Defined Data Centers (SDDC) and Software Defined Networks (SDN).

TOSCA facilitates this goal by enabling the interoperable description of application and infrastructure cloud services, the relationships between parts of the service, and the operational behavior of these services (e.g., deploy, patch, shutdown) independent of the supplier creating the service, and any particular cloud provider or hosting technology. TOSCA enables the association of that higher-level operational behavior with cloud infrastructure management.

TOSCA models integrate the collective knowledge of application and infrastructure experts, and enable the expression of application requirements independently from IaaS- and PaaS-style platform capabilities. Thus, TOSCA enables an ecosystem where cloud service providers can compete and differentiate to add value to applications in a software defined environment.

These capabilities greatly facilitate much higher levels of cloud service/solution portability, the continuous delivery of applications (DevOps) across their life cycle without lock-in, including:

- Portable deployment to any compliant cloud
- Easier migration of existing applications to the cloud
- Flexible selection and movement of applications between different cloud providers and cloud platform technologies
- Dynamic, multicloud provider applications

5.3 SNIA

The Storage Networking Industry Association (SNIA) mission is to "Lead the storage industry worldwide in developing and promoting standards, technologies, and educational services to empower organizations in the management of information".

Working towards this goal, SNIA has produced a number of specifications, of which the following have particular relevance to the SDDC.

5.3.1 Cloud Data Management Interface (CDMI)

The SNIA Cloud Data Management Interface (CDMI) is an ISO/IEC standard that enables cloud solution vendors to meet the growing need of interoperability for data stored in the cloud. The CDMI standard is applicable to all types of clouds – private, public, and hybrid. There are currently more than 20 products that meet the CDMI specification.

CDMI provides end users with the ability to control the destiny of their data and ensure hassle-free data access, data protection, and data migration from one cloud service to another.

Metadata in CDMI

The Cloud Data Management Interface (CDMI) uses many different types of metadata, including HTTP metadata, data system metadata, user metadata, and storage system metadata. To address the requirements of enterprise applications and the data managed by them, this use of metadata allows CDMI to deliver simplicity through a standard interface. CDMI leverages previous SNIA standards, such as the eXtensible Access Method (XAM), for metadata on each data element. In particular, XAM has metadata that drives retention data services useful in compliance and eDiscovery.

CDMI's use of metadata extends from individual data elements and can apply to containers of data, as well. Thus, any data placed into a container essentially inherits the data system metadata of the container into which it was placed. When creating a new container within an existing container, the new container would similarly inherit the metadata settings of its parent container. Of course, the data system metadata can be overridden at the container or individual data element level, as desired.

The extension of metadata to managing containers, not just data, enables a reduction in the number of paradigms for managing the components of storage – a significant cost savings. By supporting metadata in a cloud storage interface standard and proscribing how the storage and data system metadata is interpreted to meet the requirements of the data, the simplicity required by the cloud storage paradigm is maintained, while still addressing the requirements of enterprise applications and their data.

5.3.2 Storage Management Initiative

The SNIA's Storage Management Initiative (SMI) gathers and prioritizes industry requirements that guide the Technical Work Groups to cooperatively develop the Storage Management Initiative Specification (SMI-S), an international standard implemented in over 500 products.

SMI-S Technical Specification

SMI-S standardizes and streamlines storage management functions and features into a common set of tools that address the day-to-day tasks of the IT environment. Initially providing a foundation for identifying the attributes and properties of storage devices, SMI-S now also delivers services such as discovery, security, virtualization, performance, and fault reporting.

SMI-S defines a method for the interoperable management of a heterogeneous Storage Area Network (SAN), and describes the information available to a WBEM Client from an SMI-S compliant CIM Server and an object-oriented, XML-based, messaging-based interface designed to support the specific requirements of managing devices in and through SANs. The latest publicly released version of SMI-S is the SMI-S V1.6.1 SNIA Technical Position.

SMI-S uses the WBEM and CIM specifications from the DMTF.

5.4 Other SDOs

5.4.1 ETSI/ISG – Network Function Virtualization (NFV)

The first use case of ETSI/ISG NFV discusses NFV Infrastructure as a Service (NFVIaaS), which may have a lot of similarity with SDDC. The NFVI includes compute, networking, and storage infrastructure in virtualized forms. NFVIaaS calls for combining and interconnecting network as a service (NaaS), and other compute/storage Infrastructure as a Service (IaaS) in order to provide virtual network function (VNF) to the network administrators. The VNFs from different administrative domains can be interconnected and clustered for developing an end-to-end service. The NFV use case document is available at the following URL:

http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf.

5.4.2 IETF/IRTF

There are a few IETF and IRTF working/research groups (WGs/RGs) and drafts that discuss Virtual Data Center (VDC). The concept of VDC and the service that can be offered by using VDC are very similar to the SDDC concept that we discuss here in this paper.

The NVO3 (Network Virtualization Overlays/Over-Layer-3) Working Group (WG) focuses on developing interoperable solutions for traffic isolation, address independence, and virtual machine (VM) migration in a Data Center Virtual Private Network (DCVPN).

DCVPN is defined as a VPN that is viable across a scaling range of a few thousand VMs to several million VMs running on more than 100,000 physical servers. DCVPN supports several million endpoints and hundreds of thousands of VPNs within a single administrative domain. Further details about IETF NVO3 activities can be found at http://datatracker.ietf.org/wg/nvo3/charter/.

The SCIM (System for Cross-domain Identity Management) WG is developing the core schema and interfaces based on HTTP and REST for creating, reading, searching, modifying, and deleting user identities and identity-related objects across administrative domains.

Initial focus areas of the SCIM WG are developing a core schema definition, a set of operations for creation, modification, and deletion of users, schema discovery, read and search, bulk operations, and mapping between the inetOrgPerson LDAP object class (RFC 2798) and the SCIM schema. Further details about IETF SCIM activities can be found at http://datatracker.ietf.org/wg/scim/charter/.

The SDN (Software-Defined Networking) Research Group (RG) is currently focusing on developing definition and taxonomy for SDN. Future work may include a study of model scalability and applicability, multilayer programmability and feedback control system, network description languages, abstractions, interfaces and compilers, and security-related aspects of SDN. Further details about IRTF SDN activities can be found at https://irtf.org/sdnrg.

5.4.3 Open Networking Foundation (ONF)

ONF has developed a southbound interface (SBI; south of the controller) called OpenFlow™ to enable remote programming of the flow forwarding.

Currently ONF is focusing on Software Defined Networking (SDN) related issues especially the concepts, frameworks, and architecture.

The network segmentation, multipath and multitenancy support, and security-related activities of the Forwarding Abstraction WG, Northbound Interface (NBI) WG, Configuration and Management WG, Layer 4-7 Services DG, and Security DG may be very helpful for open SDDCs and their interconnections.

5.4.4 Open DayLight (ODL)

ODL focuses on control and programmability of the abstracted network functions and entities. The objective is to develop northbound interfaces (NBIs) for gathering network intelligence including performing analytics, and then use the controller to orchestrate adaptive new rules throughout the network for efficient automated operations. A detailed technical overview of ODL initiatives is available at http://www.opendaylight.org/project/technical-overview.

ODL supports OpenFlow and other protocols as SBIs, and released Base (Enterprise), Virtualization, and Service Provider editions of the software packages (http://www.opendaylight.org/software).

5.4.5 Open Data Center Alliance (ODCA)

ODCA initiatives and activities are focused on developing open, interoperable solutions for secure cloud federation, automation of cloud infrastructure, common management, and transparency of cloud service delivery.

6 Standards gaps - What is missing?

After we have analyzed this concept of the software defined data center and the various use cases and architectures as well as enumerating the current standards activity we realize there are several technologies that do not have well defined standards to date. This section will attempt to identify some of the key standards that will need to be explored and developed to have a truly standards based SDDC solution.

6.1 Standards for metrics

Currently there appears to be no standard metrics to be able to report and adjust resource utilization of the infrastructure and the associated application and services that are hosted upon those resources. If workloads are to be able to self-manage their required infrastructure then clearly a standard set of metrics will need to be developed. We do not have any real standard units of measure to identify both resource requirements and resource utilization.

6.2 Application and workload management

Additional work needs to be done in the instrumentation of requirements for applications and workloads. Some work has been done on deployment requirements for workloads such as specified in DMTF Open Virtualization Format (OVF) but much work still needs to be done for instrumentation of workloads and applications once they have been deployed to enable auto configuration and scaling. We also see a need for additional work for the emerging containerized applications to have their requirements be exposed in a standard way so that software defined resources may be created and removed dynamically.

6.3 Policy and service levels

To drive this level of automation there is still much work to be done in standardized policy management as well as standards to specify Service Level Objectives (SLO) that have been set based on contractual Service Level Agreements (SLA). To date work has been done on policy languages and standardized Service Level Management by organizations such as IEC/JTC1 SC38, however there is additional work to be done to create a pervasive set of standards for policy-based service level management including the standardized metrics discussed above.

7 Conclusion

To realize an SDDC, data center resources, such as compute, network, and storage, are expressed as software. They also need to have certain characteristics, such as multitenancy, rapid resource provisioning, elastic scaling, policy-driven resource management, shared infrastructure, instrumentation, and self-service, accounting, and auditing. This ultimately entails a programmable infrastructure that enables resources to be automatically cataloged, commissioned, decommissioned, repurposed, and repositioned.

8 References

S. Karavettil et al, “Security Framework for Virtualized Data Center Services”, IETF discussion draft (http://tools.ietf.org/id/draft-karavettil-vdcs-security-framework-05.txt), June 2013.

Alan G. Yoder et al, “SNIA 2015 Dictionary”, Storage Networking Industry Association, (http://www.snia.org/sites/default/files/SNIADictionaryV2015-1_0.pdf), March 2015.

SNIA Technical Community: Software Defined Storage (http://www.snia.org/sds).

Specifications

DMTF: DSP0263, *Cloud Infrastructure Management Interface (CIMI) Model and RESTful HTTP-based Protocol, version 1.1.0,October* 25 2013.
http://dmtf.org/sites/default/files/standards/documents/DSP0263_1.1.0.pdf

DMTF: *DSP0243, Open Virtualization Format Specification, version 2.1.0*, January 23 2014.
http://dmtf.org/sites/default/files/standards/documents/DSP0243_2.1.0.pdf

SNIA: *SNIA Technical Position: Cloud Data Management Interface (CDMI), v1.1.1*, March 19, 2015
http://www.snia.org/sites/default/files/CDMI_Spec_v1.1.1.pdf

SNIA: *SNIA Technical Position: Storage Management Initiative Specification (SMI-S) v1.6.1 rev 5, December 17, 2014*
http://www.snia.org/sites/default/files/SMI-Sv1.6.1r5.zip

9 Glossary

Table 1 – Glossary of terms

Acronym or Phrase	Definition	Explanation
AAA	Authentication, Authorization, and Auditing	The three major areas of concern in system security.
API	Application Programming Interface	An interface used by an application to request services. The term API is usually used to denote interfaces between applications and the software components that compose the operating environment (e.g., operating system, file system, volume manager, device drivers, etc.) Source: http://www.snia.org/education/dictionary/a
Block storage		Storage organized and allocated in blocks of fixed size.

Acronym or Phrase	Definition	Explanation
BYOD	Bring Your Own Device	The policy of permitting employees to bring personally owned mobile devices (laptops, tablets, and smart phones) to their workplace, and to use those devices to access privileged company information and applications Source: http://en.wikipedia.org/wiki/Bring_your_own_device
Cloud	Cloud Computing	Computing facilities based on remote servers accessed via internet protocols, in contrast with facilities local to their usage.
Fiber Channel		A high-speed LAN technology, most commonly used for SANs.
Firewall		A device, often implemented in software, to control data flows between two or more networks. Firewalls typically reject network traffic that does not originate from trusted address and/or ports and thus provides a degree of isolation between networks.
IaaS	Infrastructure as a Service	A delivery model for IT infrastructure whereby resources are provided as a service via network protocols. IaaS usually also provides interfaces to provision and mange resources.
IDS	Intrusion Detection System	A system used to detect unauthorized access to resources.
HIDS	Host Intrusion Detection Systems	An IDS specifically designed to protect host systems.
LAN	Local Area Network	A network with a small, restricted, scope. LAN's may be connected to larger networks, such as the internet.
Load Balancing		A mechanism used to distribute demands for resources amongst those available. Usually used in reference to processing resources but may be applied to any resource.
Metadata		Metadata is "data about data" and there are two types: structural metadata and descriptive metadata. Structural metadata is data about the containers of data. Descriptive metadata concerns the application data content.

Acronym or Phrase	Definition	Explanation
NAS	Network Attached Storage	A term used to refer to storage devices that connect to a network and provide file access services to computer systems. These devices generally consist of an engine that implements the file services, and one or more devices, on which data is stored. Source: http://www.snia.org/education/dictionary/n#network_attached_storage
NFV	Network Function Virtualization	The concept of replacing dedicated network appliances, such as routers and firewalls, with software applications running on general purpose servers.
Object storage		Storage organized and allocated as self-contained data.
PaaS	Platform as a Service	A delivery model that encapsulates underlying infrastructure to simplify developing, running, and managing applications via network protocols.
pDC	Physical Data Center	
REST	Representational State Transfer	A software architecture style consisting of guidelines and best practices for creating scalable web services. REST is a coordinated set of constraints applied to the design of components in a distributed hypermedia system that can lead to a more performant and maintainable architecture. Source: https://en.wikipedia.org/wiki/Representational_state_transfer
SaaS	Software as a Service	A delivery model whereby software applications are provided as a service via network protocols.
SAN	Storage Area Network	A network whose primary purpose is the transfer of data between computer systems and storage devices and among storage devices. Source: http://www.snia.org/education/dictionary/s#storage_area_network
SDDC	Software Defined Data Center	Refer to this document.

Acronym or Phrase	Definition	Explanation
SDN	Software Defined Network	The physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices. Source: https://www.opennetworking.org/sdn-resources/sdn-definition
SDO	Standards Development Organization	
SDS	Software Defined Storage	Virtualized storage with a service management interface. SDS includes pools of storage with data service characteristics that may be applied to meet the requirements specified through the service management interface. Source: http://www.snia.org/education/dictionary/s#software_defined_storage
Virtual Appliance		A software application preconfigured with (usually minimal) OS facilities required to run on a specific type of virtual machine. Virtual Appliances are typically used to provide services in IaaS and SaaS system architectures.
VLAN	Virtual LAN	A virtualized local area network
WAN	Wide area network	

ANNEX A
(informative)

Change log

Date	Author	Comments
2014-03-07	Hemal Shah	Initial draft
2014-04-03	Winston Bumpus	Added DMTF Standards
2014-06-13	Working Session	Merged updates and comments – Draft 9
2014-06-19	Bhumip Khasnabish	Added requirements and SDO overviews
2014-06-24	Eric Wells	Glossary & formatting
2014-09-29	Working Session	Edits for 1.0.1c
2015-05-29	Mark Carlson	Added explanatory text for the architecture diagram
2015-07-10	Eric Wells	Editorial work for WIP publication
2015-10-04		Version 1.0.0m WIP