BS ISO/IEC 5965:2021

BSI Standards Publication

Information technology — Swordfish Scalable Storage Management API Specification

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

National foreword

This British Standard is the UK implementation of ISO/IEC 5965:2021.

The UK participation in its preparation was entrusted to Technical Committee ICT/1, Information systems co-ordination.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2021
Published by BSI Standards Limited 2021

ISBN 978 0 539 15701 7

ICS 35.020

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 August 2021.

Amendments/corrigenda issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

INTERNATIONAL STANDARD

ISO/IEC 5965

First edition
2021-08

Information technology — Swordfish Scalable Storage Management API Specification

Reference number
ISO/IEC 5965:2021(E)

Documents
Document Sharing Hub
© ISO/IEC 2021

Full document details: BS ISO/IEC 5965:2021

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2021

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

© ISO/IEC 2021 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

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.

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 document should be noted (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC 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) or the IEC list of patent declarations received (see patents.iec.ch).

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

For an explanation of 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 www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by SNIA (as Swordfish Scalable Storage Management API Specification, Version 1.1.0c) and drafted in accordance with its editorial rules. It was adopted, under the JTC 1 PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Table of Contents

Table of Contents 9
1 Abstract 11
2 Scope 12
2.1 Audience Assumptions 13
3 Normative References 14
3.1 Overview 14
3.2 Approved references 14
3.3 References under development 16
3.4 Other references 16
4 Terms and Definitions 17
4.1 Overview 17
4.2 Swordfish-specific Terms 17
4.3 Reference to Redfish terms 18
4.4 Keywords (normative language terms) 19
5 Swordfish Overview 20
5.1 Introduction 20
5.2 Relation to Redfish 20
5.3 Storage System Models 21
5.4 The ServiceRoot and ServiceContainer entities 24
5.5 Swordfish model overview 25
6 Features and Profiles 28
6.1 Overview 28
6.2 Requirement for SupportedFeatures 28
6.3 EnergyStar for Storage Feature 28
6.4 Class of Service Feature 29
7 Schema Considerations 37
7.1 Schema Introduction 37
7.2 Default values and NULLABLE attributes 37
7.3 Common schema annotations 38
7.4 Property implementation requirements 39
7.5 Schema repository 40
7.6 Referencing other schemas 40
8 Implementation requirements 41
8.1 Security 41
8.2 General constraints 41
8.3 Discovering Swordfish resources 42
8.4 ClassOfService requirements 43
8.5 StorageSystems requirements 43
8.6 Entity Sets 43
8.7 Addressing entities within a collection 43
8.8 Addressing members of a ResourceCollection 44
8.9 HTTP status codes 44
9 Swordfish type definitions 48
9.1 Overview 48
9.2 Common properties 48

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

9.3 Complex Types 55
9.4 CapacitySource 1.1.2 56
9.5 ClassOfServiceCollection 63
9.6 ConsistencyGroup 1.0.1 64
9.7 ConsistencyGroupCollection 76
9.8 DataProtectionLoSCapabilities 1.1.3 78
9.9 DataSecurityLoSCapabilities 1.1.3 83
9.10 DataStorageLoSCapabilities 1.2.1 91
9.11 DriveCollection 95
9.12 EndpointGroup 1.2.0 97
9.13 EndpointGroupCollection 101
9.14 FeaturesRegistry 1.0.0 102
9.15 FileShare 1.1.3 105
9.16 FileShareCollection 111
9.17 FileSystem 1.2.2 112
9.18 FileSystemCollection 120
9.19 HostedStorageServices 121
9.20 IOConnectivityLoSCapabilities 1.1.3 122
9.21 IOPerformanceLoSCapabilities 1.1.3 126
9.22 LineOfService 1.0.0 130
9.23 LineOfServiceCollection 132
9.24 SpareResourceSet 1.0.1 133
9.25 StorageGroup 1.2.1 136
9.26 StorageGroupCollection 145
9.27 StoragePool 1.3.1 147
9.28 StoragePoolCollection 156
9.29 StorageReplicaInfo 1.3.0 157
9.30 StorageService 1.4.0 159
9.31 StorageServiceCollection 168
9.32 StorageSystemCollection 169
9.33 Volume 1.4.1 170
9.34 VolumeCollection 202
Annex A: Bibliography 205
A.1 Overview 205
A.2 Informational references 205

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

1 Abstract

The Swordfish Scalable Storage Management API ("Swordfish") defines a RESTful interface and a standardized data model to provide a scalable, customer-centric interface for managing storage and related data services. It extends the Redfish Scalable Platforms Management API Specification (DSP0266) from the DMTF.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

2 Scope

Swordfish extends the Redfish Scalable Platforms Management API Specification, as defined by ISO 30115 It defines a comprehensive, RESTful API for storage management that addresses block storage, file systems, object storage, and storage network infrastructure. It is centered around common operational and business concerns of storage management, including:

- Configuration and provisioning
- Monitoring
- Event and log management
- Performance assessment
- Diagnostics
- Fault detection and remediation
- Security
- Accounting and resource consumption

Swordfish's storage model is built around well-defined classes of service, which provide a means to map high-level business goals and objectives to specific, storage-based actions and requirements, in a clear and consistent way that can be applied uniformly across a broad spectrum of storage configurations and storage types (e.g., block storage, file systems, object stores). Common storage management functionality covered by class of service includes snapshots, replication, mapping and masking, and provisioning.

The Redfish specification provides the protocols and a core set of data models and behaviors for the management of systems. It defines the elements and behaviors that are mandatory for all Redfish implementations. Additionally it defines additional elements and behaviors that can be chosen by system vendors or manufacturers. The specifications also defines points at which OEM (system vendor) extensions can be provided by a given implementation. The specifications specifies normative requirements for Redfish Services and associated materials, such as Redfish Schema files. The Redfish specifications does not set requirements for Redfish clients, but will indicate what a Redfish client should do in order to access and utilize a Redfish Service successfully and effectively.

The Swordfish specification defines additional data models and behaviors for the management of storage systems and storage infrastructure. A Swordfish implementation shall conform to all requirements specified in the Redfish specifications.

Swordfish is suitable for a wide range of storage, from small-scale object drives, integrated RAID cards or RBODs providing storage services, to external disk arrays or file servers, to infrastructure providing storage services for converged, hyperscale and large scale cloud environments.

This document defines the Swordfish Scalable Storage Management API.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

2.1 Audience Assumptions

As Swordfish is designed as an extension of the Redfish specification, this document is written with the presumption that the reader has a detailed understanding of ISO 30115 and any updates or clarifications introduced by the DMTF. This document cannot be fully understood without that context.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

3 Normative References

3.1 Overview

The documents listed in Table 3 is indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

3.2 Approved references

Table 3: Approved normative references

Tag	Title (Version)	Author	URL
ISO-8601	Data elements and interchange formats – Information interchange – Representation of dates and times – Part 1: Basic rules	ISO/IEC	http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=70907

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Table 3: Approved normative references, cont.

Tag	Title (Version)	Author	URL
ISO-Direct	ISO/IEC Directives, Part 2 Principles and rules for the structure and drafting of ISO and IEC documents (Seventh Edition, 2016)	ISO/IEC	http://isotc.iso.org/livelink/livelink/fetch/2000/2122/4230450/4230456/ISO_IEC_Directives *Part_2 Principles_and_rules_for_the structure_and_drafting_of_ISO_and_IEC* documents *-2016%287th_edition%29-*_PDF.pdf?nodeid=17667902&vernum=-2
OData	Open Data Protocol (v. 4.0)	OASIS	https://www.oasis-open.org/standards#odatav4.0
RFC3986	Uniform Resource Identifier (URI): Generic Syntax (2005)	The Internet Society	http://www.rfc-base.org/txt/rfc-3986.txt
CSDL	Common Schema Definition Language (4.0)	OASIS	http://docs.oasis-open.org/odata/odata/v4.0/odata-v4.0-part3-csdl.html
ITIL	ITIL Glossary (2011)	ITIL	https://www.axelos.com/Corporate/media/Files/Glossaries/ITIL_2011_Glossary_GB-v1-0.pdf
Units	The Unified Code for Units of Measure (v2.0.1)	Regenstrief Institute, Inc. and the UCUM Organization	http://unitsofmeasure.org/trac
SPC-4	SCSI Primary Commands - 4 (SPC-4) INCITS 513-2015	T10	http://www.techstreet.com/cgi-bin/joint.cgi/incits
Features	Swordfish Features Registry, version 1.0.1	SNIA	https://redfish.dmtf.org/registries/swordfish/v1/SwordfishFeatureRegistry.1.0.1.json

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Table 3: Approved normative references, cont.

Tag	Title (Version)	Author	URL
Messages	Swordfish Message Registry, version 1.0.2	SNIA	https://redfish.dmtf.org/registries/swordfish/v1/Swordfish.1.0.2.json
EnergyStar	ENERGY STAR Data Center Storage Version 1.1 Updated Program Requirements – April 1, 2019	EPA	https://www.energystar.gov/sites/default/files/ENERGY STAR Data Center Storage Final Version 1.1 Specification Rev. April 2019.pdf
ISO-20648	Information technology — TLS specification for storage systemsi	ISO/IEC	https://www.iso.org/standard/68622.html
ISO-30115	ISO/IEC 30115(en) Information technology — Redfish scalable platforms management API specification	ISO/IEC	https://www.iso.org/standard/53235.html

3.3 References under development

None defined in this document.

3.4 Other references

None defined in this document.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

4 Terms and Definitions

4.1 Overview

In this document, some terms have a specific meaning beyond the normal English meaning. Those terms are defined in this clause. New terms, frequently used Redfish terms.

4.2 Swordfish-specific Terms

4.2.1 Definitions

Table 4 summarizes the terms are used in this document.

Table 4: Swordfish terms

Term	Definition
Entity	An instance of a schema element.
Model	A set of entities and the relationships between them that define the semantics, behavior and state of that set.
OData service	A REST-based service that allows resources, identified using Uniform Resource Locators (URLs) and defined in a model, to be published and edited by Web clients using simple HTTP messages.
Resource	A central element in a model, which represents a physical construct or a logical service, and is further defined by other model entities.
Schema	A formal language representation of a model that conforms to a metamodel.
Service Document	A particular resource that is directly accessed via an OData service entry point. This resource serves as a starting point for locating and accessing the other resources and associated metadata that together make up an instance of a Swordfish service.
Swordfish service	An extension to the Redfish Service that conforms to the Swordfish specification, and provides REST-ful storage management functionality.

4.2.2 Symbols and abbreviated terms

None in this document.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

4.3 Reference to Redfish terms

Many terms in this document were originally defined in the Redfish Specification. Some of the more common terms and definitions are reproduced in Table 5, as an aid to the reader.

Table 5: Redfish terms

Term	Definition (as of 16 August 2019)
OData	The Open Data Protocol, as defined in OData-Protocol.
OData Service Document	Resource that provides information about the service root for generic OData clients.
Redfish Schema	Defines Redfish Resources according to OData schema representation. You can directly translate a Redfish Schema to a JSON Schema representation.
Redfish service	Implementation of the protocols, resources, and functions that deliver the interface that this specification defines and its associated behaviors for one or more managed systems.
Request	A message from a client to a service.
Service Root	Resource that serves as the starting point for locating and accessing the other resources and associated metadata that together make up an instance of a Redfish Service.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

4.4 Keywords (normative language terms)

This document conforms to ISO/IEC Directives, Part 2 for keyword usage. The most common terms and their intended meanings are summarized in Table 6.

Table 6: Normative language terms

Term(s)	Meaning
shall / shall not	Used to identify objectively verifiable criteria to be fulfilled and from which no deviation is permitted if compliance with the document is to be claimed
should / should not	Used to identify a suggested possible choice or course of action deemed to be particularly suitable without necessarily mentioning or excluding others
may / need not	Used to convey consent or liberty (or opportunity) to do something
can / cannot	Expected or conceivable material, physical or causal outcome
must	Identifies a constraint or obligation on the user of the document, typically due to one or more legal requirements or laws of nature, that is not stated as a provision of the standard *NB:* "must" is not an alternative for "shall", and should only be used for constraints that arise from outside this standard

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

5 Swordfish Overview

5.1 Introduction

The Swordfish Scalable Storage Management API ("Swordfish") defines a RESTful interface and a standardized data model to provide a scalable, customer-centric interface for managing storage and related data services. It extends the Redfish Scalable Platforms Management API Specification (DSP0266) from the DMTF.

5.2 Relation to Redfish

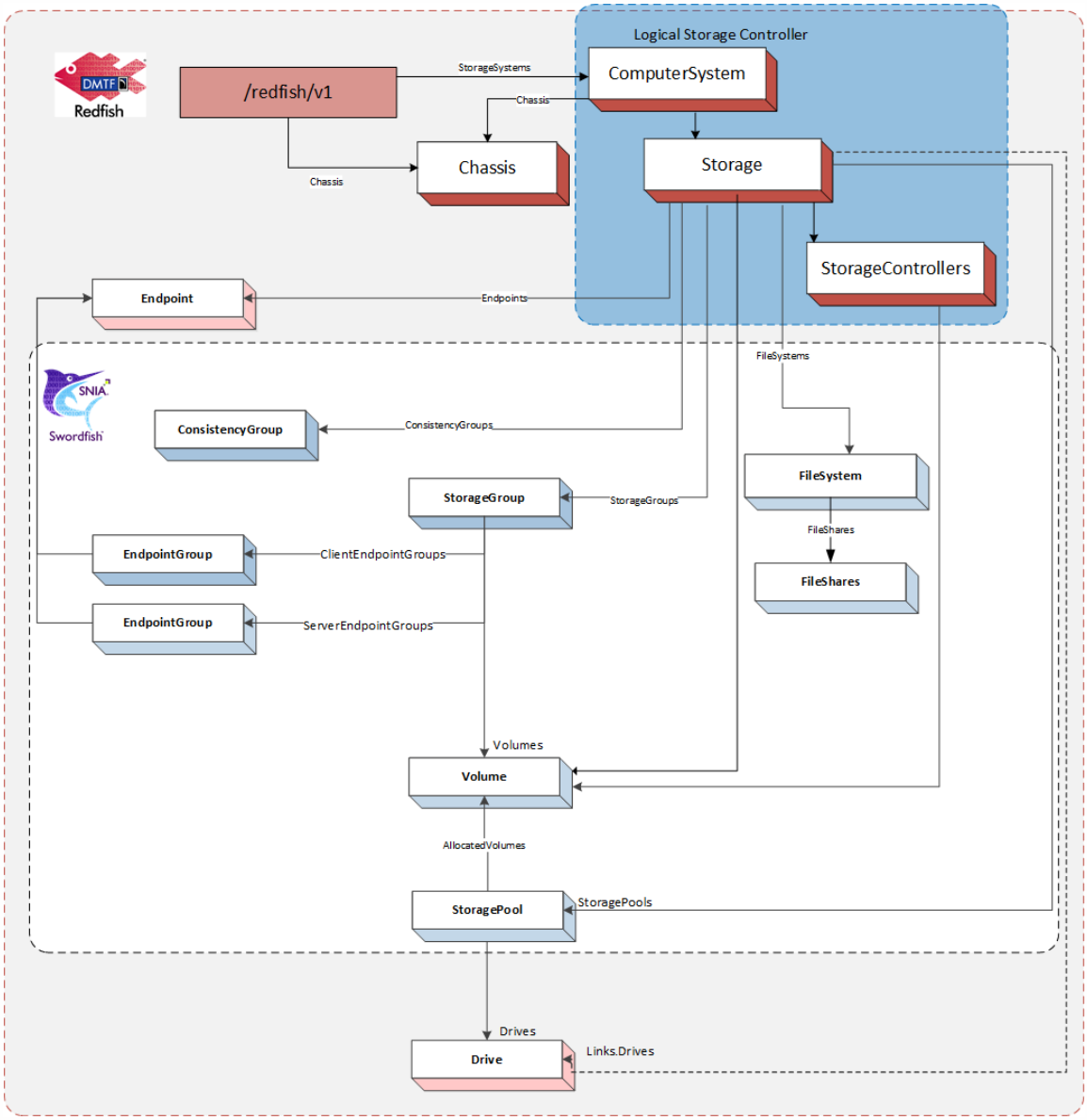

Figure 1: Extension Overview

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

The Swordfish service interface extends the Redfish service interface, as illustrated in Figure 1. As such, a Swordfish service is a Redfish service and includes all required elements of the Redfish model.

Storage systems managed by the Swordfish service are located in the `ServiceRoot` via the `StorageSystems` resource collection. They are modeled using Redfish `ComputerSystems`. The physical infrastructure is modeled using Redfish `Chassis`.

As modeling for storage systems may cover both logical and physical constructs, Swordfish management clients that are focused on logical storage management use cases may choose to manage functionality entirely by way of logical resources.

Each Swordfish service is accessed via well known URLs on the system supporting the Swordfish Service. Since Swordfish is an extension of Redfish, these URLs are the same as for accessing the Redfish defined aspects of the service.

5.3 Storage System Models

Swordfish has been designed to support a broad range of configurations, requirements, size and complexity, as well as logical and physical architectures. As a result, there are two primary methods of modelling the storage system for a Swordfish implementation:

1. Swordfish Integrated Configuration

The SIC uses the same ComputerSystem model instantiation as the server where the physical element resides.

The logical storage controller is modeled using the Redfish `Storage` and `StorageController` resources. The `Storage` resource is located in the Redfish hierarchy contained by `ComputerSystems`, typically running as ApplicationServers. The physical infrastructure is modeled using Redfish `Chassis`. Managed resources are connected to the `Storage` resource, including `Volumes` the `StoragePools`. These relationships are illustrated in Figure 2.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

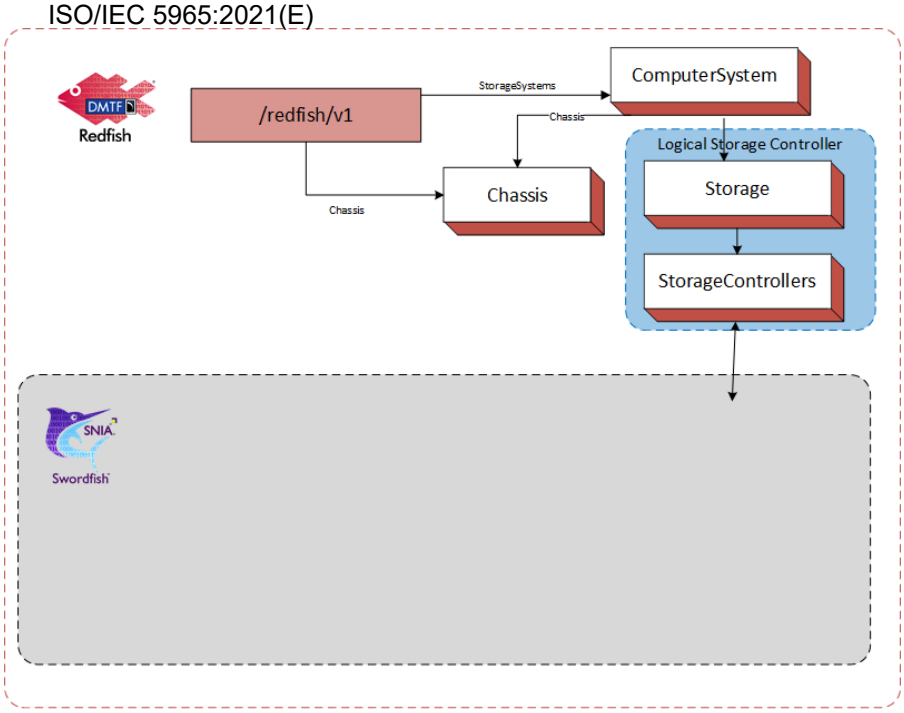

Figure 2: Integrated Configuration Overview

This configuration works well when the storage system can be modeled by simply instantiating a new Storage object within an existing computer system. An example of a Storage System for an integrated configuration is shown in Figure 3.

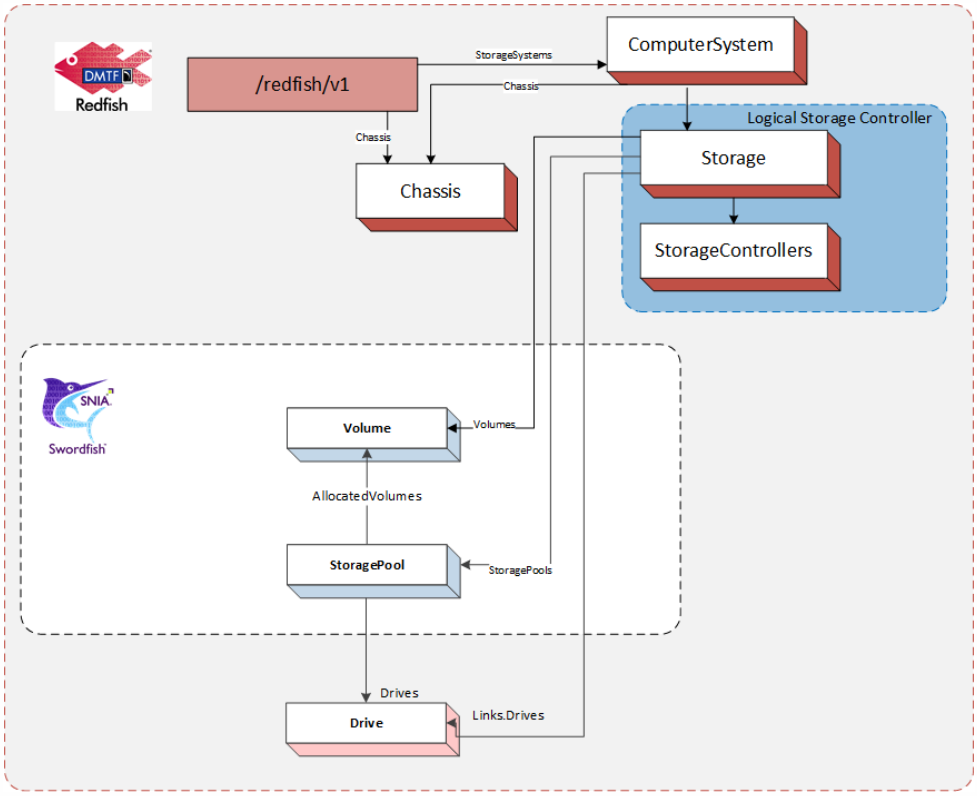

Figure 3: Integrated Configuration Example

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

2. Swordfish Standalone Configuration

The SSC uses separate ComputerSystem/StorageSystem model instantiation(s) to represent/model the logical controller(s) for the system.

The logical storage controller is modeled using Redfish a `ComputerSystems` with properties set as a `StorageSystem`. The physical infrastructure is modeled using Redfish `Chassis`. Managed resources are then connected to the `Storage` resource, including `Volumes`, `StoragePools`, `ConsistencyGroups`, and `StorageGroups`. These relationships are illustrated in Figure 4.

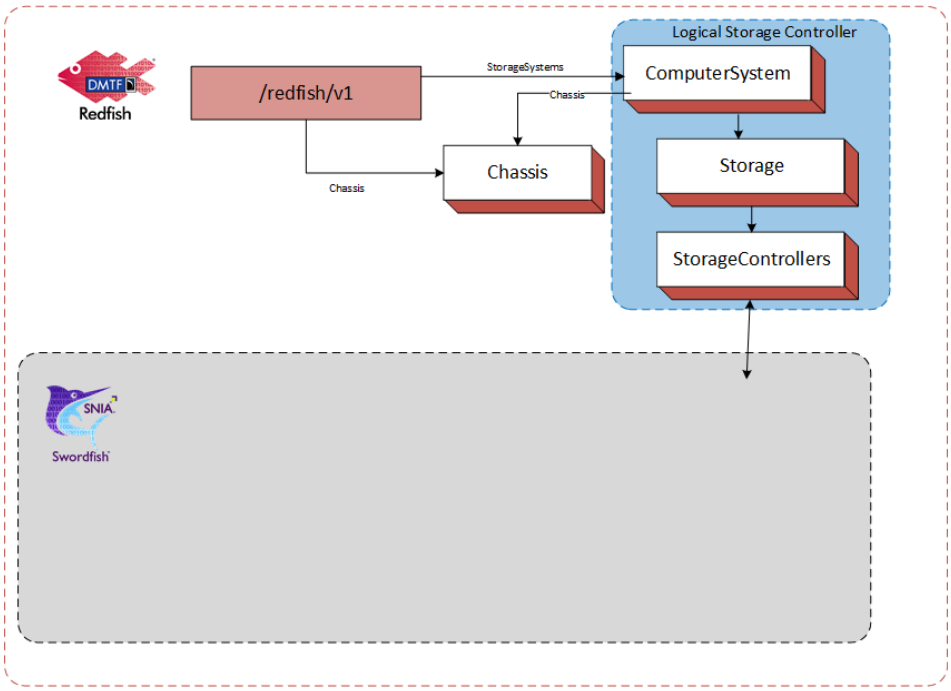

Figure 4: Standalone Configuration Overview

This configuration works well when the storage system needs a new `ComputerSystem` instance to model the logical controller. An example of a Storage System for a hosted service configuration is shown in Figure 5.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

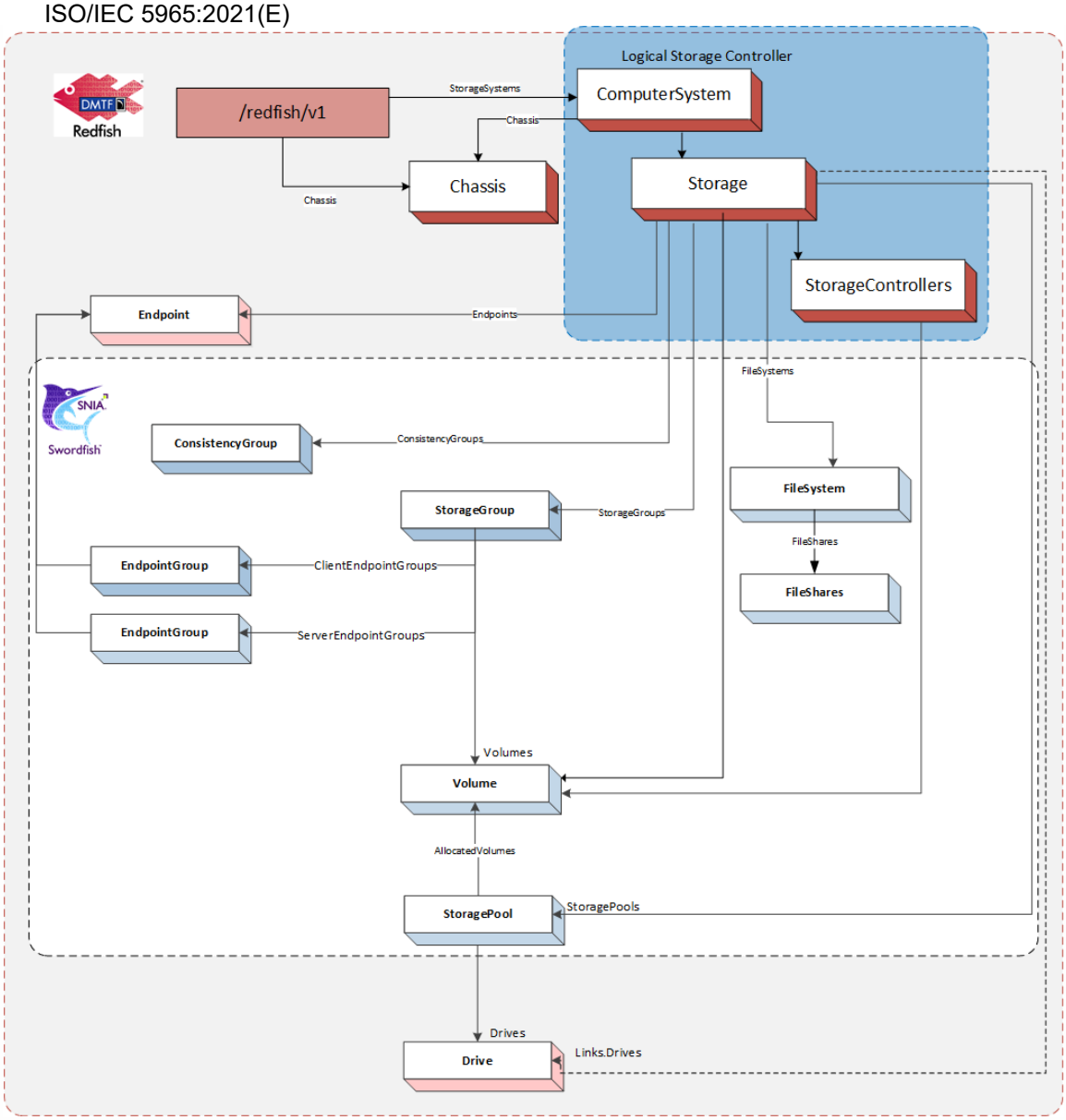

Figure 5: Standalone Configuration Example

5.4 The ServiceRoot and ServiceContainer entities

5.4.1 Overview

A **GET** of `/redfish/v1` will return the `ServiceRoot` entity. A **GET** of `/redfish/v1/odata` will return the `ServiceContainer` instances that represents the OData service document. Each of these instances provides links to the remainder of the system.

The following are the elements utilized for Swordfish management.

- `Systems`: A reference to a Systems resource collection;
- `Chassis`: A reference to a Chassis resource collection;

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

- `StorageSystems`: A reference to a StorageSystems resource collection.

5.4.2 The Systems resource collection

A resource collection that references a set of `ComputerSystem` resources that each represents a general purpose application server. Each `ComputerSystem` resource will have an entry with the value of "ApplicationServer" in its `HostingRoles` property. A particular `ComputerSystem` resource can be in both the StorageSystems collection and the Systems collection.

5.4.3 The Chassis resource collection

A resource collection that references a set of `Chassis` resources. Each Chassis resource represents physical containers, (i.e. sheet-metal confined spaces and logical zones like racks, enclosures, chassis and all other containers). Subsystems (like sensors), which operate outside of a system's data plane (meaning the resources are not accessible to software running on the system) are linked either directly or indirectly through this resource.

5.4.4 The StorageSystems resource collection

A reference to a `ComputerSystemCollection` with members of type `ComputerSystem` that support storage services. These `ComputerSystem` resources represent systems that support Swordfish storage management services. They will have an entry with the value of "StorageServer" in their `HostingRoles` property.A resource collection that references a set of `ComputerSystem` resources that each represents a storage server. Each `ComputerSystem` resource will have an entry with the value of "StorageServer" in its `HostingRoles` property. A particular `ComputerSystem` resource can be a member of both the `StorageSystems` resource collection and the `Systems` resource collection.

5.5 Swordfish model overview

5.5.1 The Storage resource

The storage system exposes logical storage, associated resources and related functionality. Storage service resources can be found in the service root or service container via the `StorageSystem` resource collection, and are attached to the `Storage` object within the `StorageSystem` (`ComputerSystem`).

The following are the principal properties of `Storage` that point to resources managed or defined by the storage system:

- `Drives`: A reference to a collection that collects `Drive` resources used for storage.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

- `Enclosures`: A reference to a resource collection that collects Chassis resources that contain storage related resources.
- `Endpoints`: A reference to a resource collection that collects `Endpoint` resources used to access storage.
- `EndpointGroups`: A reference to a resource collection that collects `EndpointGroups` resources.
- `FileSystems`: A reference to a resource collection that collects `FileSystem` resources.
- `StorageGroups`: A reference to a resource collection that collects `StorageGroup` resources.
- `ConsistencyGroups`: A reference to a resource collection that collects `ConsistencyGroup` resources.
- `StoragePools`: A reference to a resource collection that collects `StorageGroup` resources.
- `Volumes`: A reference to a resource collection that collects `Volume` resources.

5.5.1.1 The Endpoint resource

Endpoints represent one end of a protocol specific connection that supports sending or receiving messages according to a particular protocol.

5.5.1.2 The Endpoint Collection resource

The `Endpoint` Group is resource collection that references a set of `Endpoint` resources.

5.5.1.3 The ConsistencyGroup resource

ConsistencyGroups represent a set of volumes that are managed consistently and collectively as a group, to allow system and application level activities to be performed on a set of data that spans volumes. This activities include device-level replication activities as well as system level functions, such as reset.

When ConsistencyGroups are implemented, they are attached to a `Storage` resource and its internal `Volumes` collection is constructed from a subset of the `Volumes` collection of the `Storage` resource.

5.5.1.4 The ConsistencyGroup Collection resource

The `ConsistencyGroupCollection` is a resource collection that references a set of `ConsistencyGroup` resources.

5.5.1.5 The StorageGroup resource

StorageGroups represent a set of volumes that are managed as a group in order to facilitate mapping and masking, in which the volumes of a storage group are collectively exposed or hidden to a set of clients.

The set of volumes is specified by the `Mapped Volumes` attribute. `MappedVolumes` is a resource collection of the `Mapped Volume` construct (a tuple of a pointer to a volume and a corresponding Logical Unit Number for that volume).

AVA Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

The set of client endpoints to which the volumes can be exposed is specified by the `ClientEndpointGroups`attribute. The `ClientEndpointGroup` resource specifies a collection of `EndpointGroup` resources.

The set of server endpoints to which the volumes can be exposed is specified by the `ServerEndpointGroups`attribute. The `ServerEndpointGroup` resource specifies a collection of `EndpointGroup` resources.

5.5.1.6 The StoragePool resource

The `StoragePool` resource represents unassigned storage capacity that can be used to produce storage volumes or other storage pools.

The following are the principal properties of `StoragePool` that are used to identify resources provisioned or supported by the storage pool:

- `AllocatedVolumes`: A reference to a resource collection that collects `Volume` resources that have been provisioned from the storage pool.
- `AllocatedPools`: A reference to a resource collection that collects `StoragePool` resources that have been provisioned from the storage pool.

5.5.1.7 The Volume resource

`Volume` resource represents a block-addressable container of storage, sometimes referred to as a “Logical Unit”, “LU”, “LUN”, or “StorageVolume” in the storage industry.

5.5.1.8 The FileSystem resource

This `FileSystem` resource represents a file system. Each `FileSystem` may contain a collection of `FileShares` that can be presented to hosts.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

6 Features and Profiles

6.1 Overview

Features are high-level descriptions of functionality which an implementation uses to advertise what functionality it currently supports, and for some features, is capable of supporting.

The detailed definitions required to describe to implementers how to implement a feature are written in profile definition files. A feature is generally represented in one (but may be more) profile definition file, or profile.

Profiles are detailed descriptions that describe down to the individual property level what functionality is required in order to advertise features. Different profile definitions can exist for the same feature type but for various types of storage configurations: Swordfish.Block.Provisioning, Swordfish.File.Provisioning

The Swordfish Features Registry shall be used to advertise what standard and Oem Features an implementation supports.

6.2 Requirement for SupportedFeatures

`SupportedFeatures` entries in the `Features` registry represent the client's primary initial runtime view of the capabilities of a Swordfish implementation. Without properly formed entries in this registry, there is no visibility to an implementation's functionality.

Swordfish implementations shall implement the `Features` registry and advertise at least the `SNIA.Swordfish.Discovery` supported feature in order to be considered a Swordfish implementation.

Features define coarse-grained sets of functionality. In order to advertise a feature (using the `SupportedFeature` mechanism in the SupportedFeatures Registry), the implementation must support the complete set of functionality as defined in the corresponding profile.

The Swordfish Features Registry publishes the official list of supported SNIA Features, and provides a high-level description of their functionality. Many of those features are self-explanatory (e.g., local replication, remote replication), but there are some features where additional context is appropriate:

- Class of Service
- Energy Star for Storage

6.3 EnergyStar for Storage Feature

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

The EnergyStar for Storage Feature and profile has been created to formalize the requirements from the ENERGY STAR Data Center Storage Program Requirements on storage products. The profile indicates what properties Swordfish implementations need to support in order to properly instrument EnergyStar reporting capability. This functionality is intended to support EnergyStar data gathering requirements as part of the EnergyStar certification process.

6.4 Class of Service Feature

6.4.1 Overview

Swordfish supports a `ClassOfService` feature. The `ClassOfService` functionality supports systems that are capable of providing a greater level of management automation, where a higher-level set of goals is provided as direction rather than requiring parameterized inputs for all configuration actions.

The Class of Service feature uses a combination of device-defined capabilities to structure LinesOfService, which are sets of available functionality in a given system, that can then be grouped together to provide classes of service.

When Class of service functionality is implemented, the Swordfish functionality may be entirely exposed through the StorageService resource. Each Swordfish StorageService is located in the ServiceRoot (and ServiceContainer) via the StorageServices resource collection.

6.4.2 Class of Service Model

For Swordfish with a class of service interface, the following two models apply. Either model choice results in the same storage service, regardless of the storage system model.

1. Integrated Service Configuration

The storage systems managed by the Swordfish storage service are modeled using the Redfish `Storage` resource and `StorageController` resource collections. The `Storage` resource is located in the Redfish hierarchy contained by `ComputerSystems`, typically running as ApplicationServers. The physical infrastructure is modeled using Redfish `Chassis`. These relationships are shown in Figure 6.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

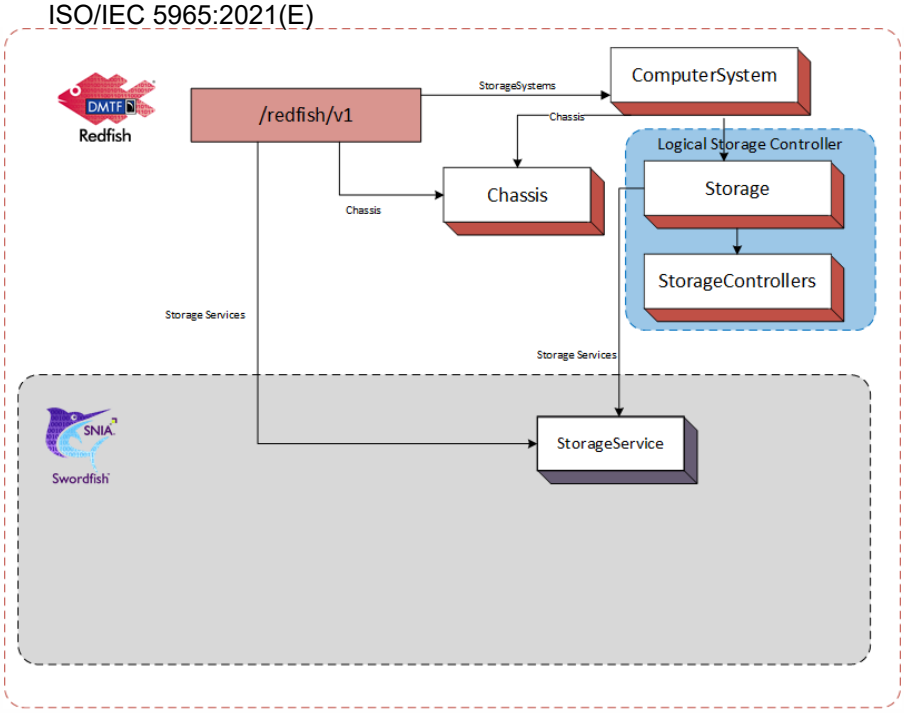

Figure 6: Integrated Service Overview

This configuration works well when the storage service is hosted by a storage resource within a computer system. An example of a Storage Service for an integrated service configuration is shown in Figure 7.

> Note: This diagram and the discussion of the configuration description have been simplified slightly to avoid confusion. A full implementation would likely include additional links to the logical storage controller resources.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

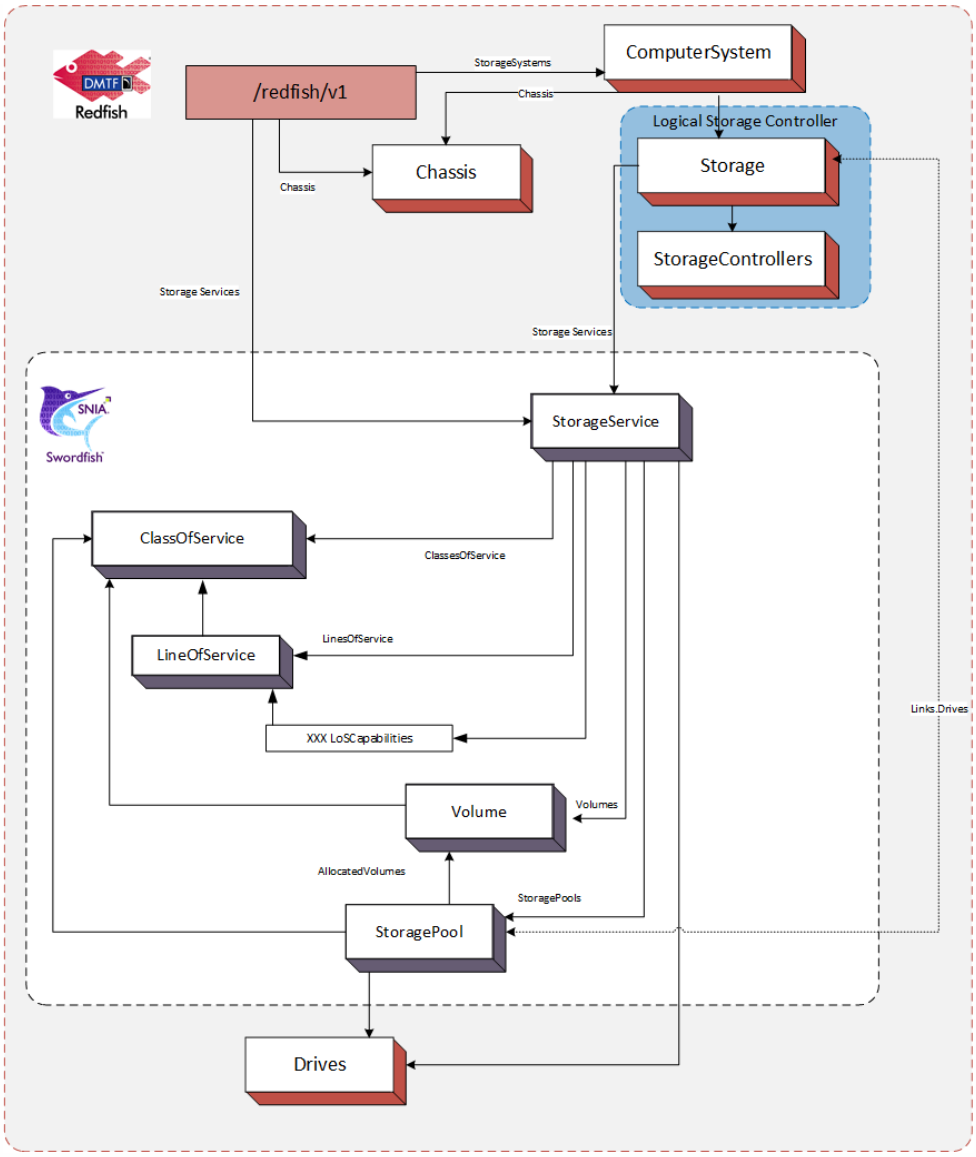

Figure 7: Integrated Service Example

2. Hosted Service Configuration

The storage systems managed by the Swordfish storage service are located in the `ServiceRoot` (and `ServiceContainer`) via the `StorageSystems` resource collection. They are modeled using Redfish `ComputerSystems`. The physical infrastructure is modeled using Redfish `Chassis`. These relationships are shown in Figure 8.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

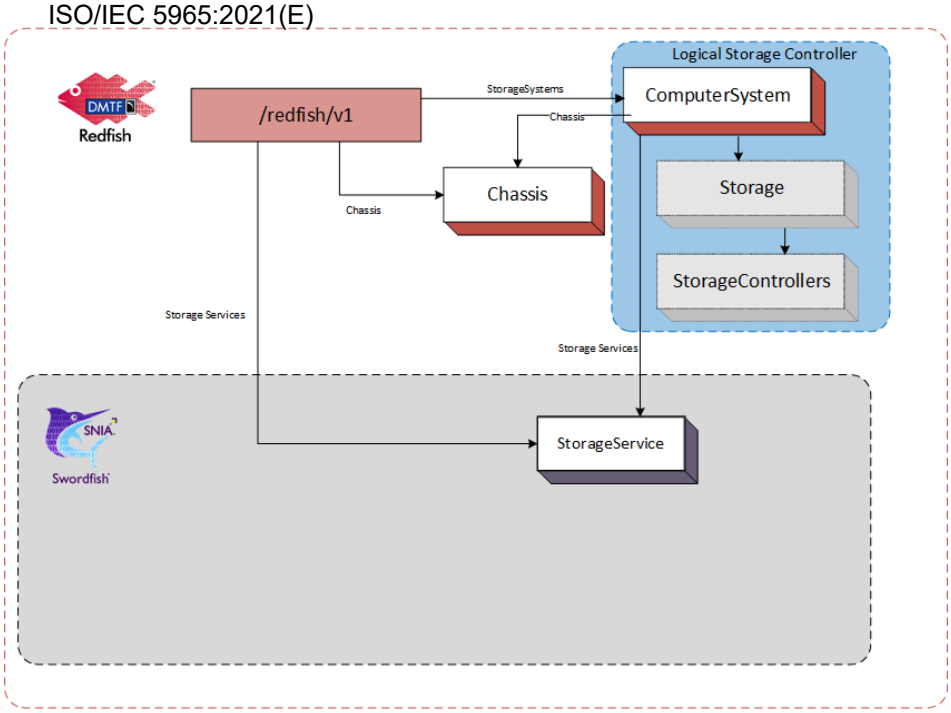

Figure 8: Hosted Service Overview

This configuration works well when the storage system hosts the storage service directly. An example of a Storage Service for a hosted service configuration is shown in Figure 9.

> Note: This diagram and the discussion of the configuration description have been simplified slightly to avoid confusion. A full implementation would likely include additional links to the logical storage controller resources.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

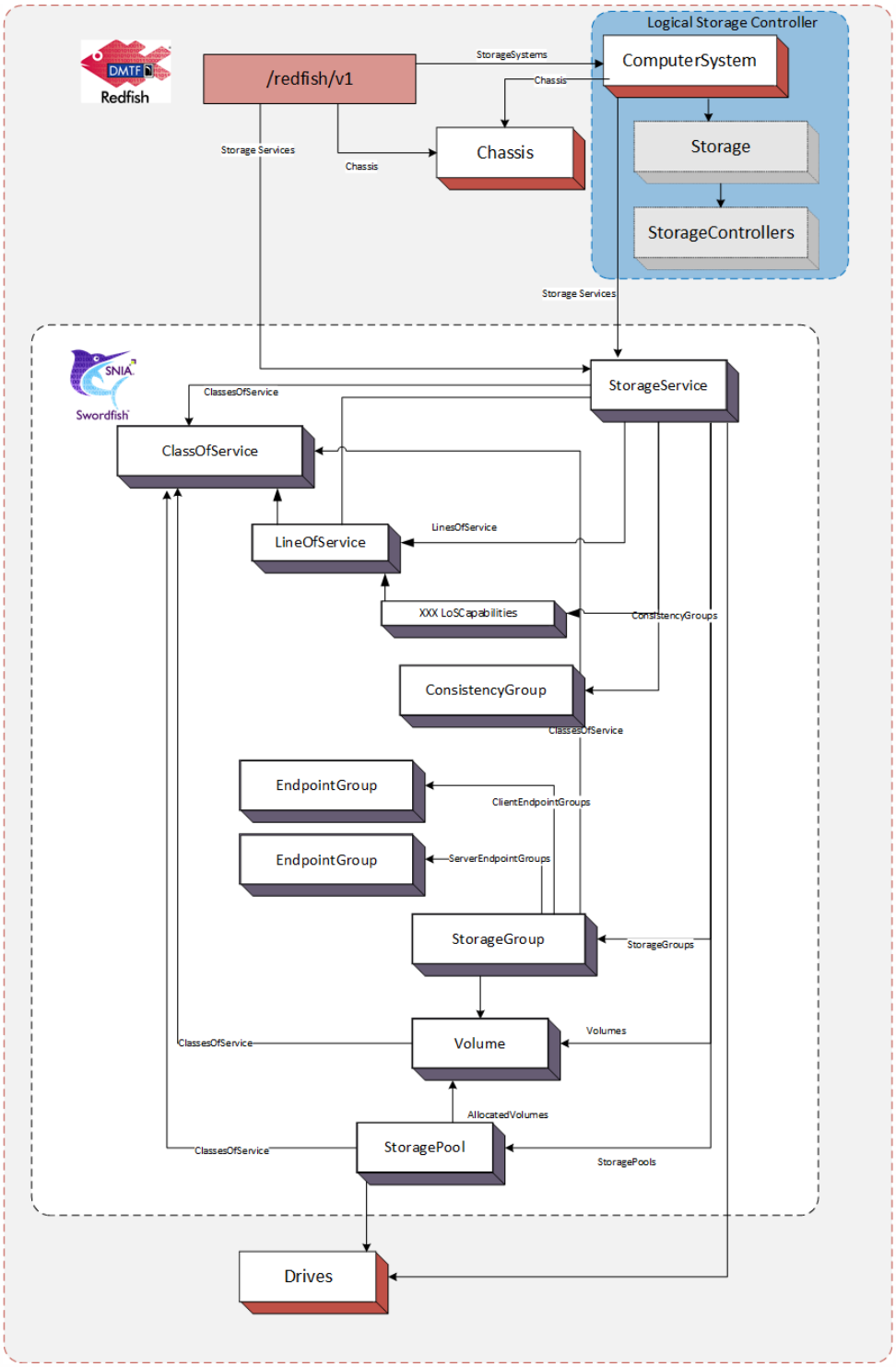

Figure 9: Hosted Service Example

6.4.3 ServiceRoot Additions

When the StorageService feature is implemented, the following is added to the `ServiceRoot`:

- `StorageServices`: A resource collection that references a set of `StorageService` resources. Each `StorageService` resource represents the resources and behaviors supported by that storage service.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

6.4.4 The StorageService resource

6.4.4.1 Principal Properties

The storage service is hosted on a storage system and exposes logical storage, associated resources and related functionality. Storage service resources can be found in the service root or service container via the `StorageServices` resource collection.

The following are the principal properties of `StorageService` that point to resources managed or defined by the storage service:

- `ClassesOfService`: A reference to a resource collection that specifies the supported `ClassOfService` resources.
- `Drives`: A reference to a resource collection that collects `Drive` resources used for storage.
- `Enclosures`: A reference to a resource collection that collects Chassis resources that contain storage related resources.
- `Endpoints`: A reference to a resource collection that collects`Endpoint` resources used to access storage.
- `FileSystems`: A reference to a resource collection that collects `FileSystem` resources.
- `EndpointGroups`: A reference to a resource collection that collects `EndpointGroups` resources.
- `StorageGroups`: A reference to a resource collection that collects `StorageGroup` resources.
- `StoragePools`: A reference to a resource collection that collects `StorageGroup` resources.
- `Volumes`: A reference to a resource collection that collects `Volume` resources.
- `HostingSystem`: A reference to the `ComputerSystem` instance that hosts this `StorageService`.

6.4.4.2 Capabilities and Lines of ServiceRoot

The following properties each define a set of attributes, which describe capabilities that the storage service may support:

- `DataProtectionLoSCapabilities`: Replicas that protects data from loss.
- `DataSecurityLoSCapabilities`: Data security service level requirements. The data security characteristics enable the storage system to be used in an environment where compliance with an externally-specified security standard or standards is required. Examples of such standards include FIPS-140, HIPAA and PCI.
- `DataStorageLoSCapabilities`: Provisioning and access characteristics for storage of the data.
- `IOConnectivityLoSCapabilities`: IO connectivity requirements for access to the data.
- `IOPerformanceLoSCapabilities`: IO performance requirements for access to the data.

In each of the above, not all combinations of attribute values defined within a capability are likely to be supported by the storage service.

Known, supported combinations of attribute values are used to construct entries in the `LinesOfService` array property. Not all attributes of a line of service entry need be specified (i.e. some may be Null). If an attribute has no value, the storage service may choose any supported values when provisioning for that entry. Otherwise, the line of service attribute values specifies the kind or level of service to be provided.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

6.4.4.3 The ClassOfService resource

A class of service represents a choice of utility or warranty offered to customers by a service. (ITIL uses the term `service option`. See the Normative References.)

Each `ClassOfService` resource is a uniquely named description of the characteristics of one choice of utility or warranty for a service. Each `ClassOfService` is a description of the kind and quality of service to provide and is not intended to describe how the service provides that service.

Each `ClassOfService` is defined by an aggregation of lines of service. Supported lines of service are listed in the corresponding capabilities attributes of the storage service, (see above).

Currently defined lines of service are:

- Data Protection: Describes the characteristics of a replica that protects data from loss.
- Data Security: Describe data security service level requirements. The data security characteristics enable the storage system to be used in an environment where compliance with an externally-specified security standard or standards is required. Examples of such standards include FIPS-140, HIPAA and PCI.
- Data Storage: Describes provisioning and access characteristics for storage of the data.
- IO Connectivity: Describes IO connectivity requirements for access to the data.
- IO Performance: Describes the IO performance requirements for access to the data under a particular workload.

Some advertised `ClassOfService` resources are created by the service implementation. These are generally not changeable and are intrinsic to the implementation.

A service may support creation or modification of `ClassOfService` resources. All must be consistent with the capabilities of the service.

6.4.4.4 The StoragePool resource

When a Swordfish implementation advertises support for the Class of Service feature, the `StoragePool` resource now presents a new method to the client to allocate unassigned storage capacity. This is automated by the system as conformance to one or more classes of service. Requests to `StoragePool` shall automatically allocate capacity based on the constraints of the selected class of service and any other selected parameters, with priority given to the class of service constraints.

The following are the principal properties of `StoragePool` that are used to identify resources provisioned or supported by the storage pool related to Class of Storage:

- `ClassesOfService`: A reference to a resource collection that specifies the set `ClassOfService` resources that can be specified when provisioning resources from the storage pool.
- `DefaultClassOfService`: A reference to the default `ClassOfService` resources used for provisioning from the storage pool.

6.4.4.5 The Volume resource

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

`Volume` resource represents a block-addressable container of storage, sometimes referred to as a "Logical Unit", "LU", "LUN", or "StorageVolume" in the storage industry. Volumes optionally adhere to a `ClassOfService`, which defines added functionality. Examples include:

- Access capabilities
- Capacity and capacity sources
- Consumption tracking (e.g., LowSpaceWarningThresholdPercents)
- Replication details
- StorageGroup Information

6.4.4.6 The FileSystem resource

In a Swordfish implementation that advertises support for the Class of Service feature, File systems represent file-addressable capacity that are conformant to a `ClassOfService`.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

7 Schema Considerations

7.1 Schema Introduction

7.1.1 Overview

A Swordfish implementation is a Redfish implementation, and as such it minimally includes support for some Redfish-defined schema, including ServiceRoot and ComputerSystem. Swordfish implementations include support for Swordfish-defined schema. The Swordfish model focuses primarily on the logical model of a storage system and does not require full representation of a physical instantiation. This is left to the implementer to complete from available Redfish schema models.

Swordfish schema is conformant with the rules used to define Redfish schema. Redfish schema is conformant with the Common Schema Definition Language, see CSDL. This section provides additional definition and context for the CSDL elements used to define Swordfish schema.

7.1.2 Swordfish Extension of the Redfish ServiceRoot

The Redfish `ServiceRoot` has properties that provide access to Swordfish resources.

The first is `StorageSystems`. This property references a collection of `ComputerSystem` resources that each support Swordfish functionality. Each such `ComputerSystem` shall have an entry in its `HostingRoles` property with the value of `StorageServer`.

For implementations that advertise support for the `ClassOfService` feature, the implementation shall instantiate a collection of `StorageServicesat` the `ServiceRoot` with at least one member. The collection provides the client an efficient means to search across all `StorageService` resources, regardless of which `ComputerSystem` is supporting the service.

7.2 Default values and NULLABLE attributes

The interaction of `Nullable` and `DefaultValue` needs to be clearly understood by both implementers and client developers. The possible combinations of are summarized in Table 7. The table contains:

- **Nullable**: True, if a given property may be NULL
- **DefaultValue**: True, if a default value is provided for a given property
- **Client**: True, if a client value is supplied for a given property in a query or response
- **Result**: The resultant value of the given property. One of:

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

- *C*: The client-provided value
- *D*: The default value
- *Null*: Null
- *I*: Implementation defined
- *Error*: Error state

Table 7: Default and Nullable Interaction

Nullable	DefaultValue	Client	Value
T	T	T	C
T	T	F	D
T	F	T	C
T	F	F	I or Null
F	T	T	C
F	T	F	D
F	F	T	C
F	F	F	I or Error

7.3 Common schema annotations

Table 8 lists common annotation used in the definition of Swordfish, for details see OData Capabilities Vocabulary, OData Core Vocabulary, OData Measures Vocabulary, and Redfish Extensions,

Table 8: Schema annotations

Name	Applies to	Description
AllowableValues	Parameter	The set of allowable values for a parameter
AutoExpand	NavigationProperty	If true, return expand the target element
AutoExpandReferences	NavigationProperty	If true, return references to the target element

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021

Table 8: Schema annotations, cont.

Name	Applies to	Description
ConformanceLevel	EntityContainer	Specifies OData conformance level
Deprecated	All	Specifies that the element may be removed in future major revisions, but shall continue to be supported as specified in the current revision.
Description	All	A brief description of a model element
LongDescription	All	A normative description of a model element
Maximum	Parameter, Property	Maximum value that an integer property or parameter may have
Minimum	Parameter, Property	Minimum value that an integer property or parameter may have
Pattern	Parameter, Property	Specifies a pattern that the value shall match
Permissions	NavigationProperty, Property	Access permission for the property.
Required	NavigationProperty, Property	If true, property is required to be supported by the service. The default is optional. See *Required Properties*
RequiredOnCreate	NavigationProperty, Property	If true, property is required on creation. See *Required Properties*
Unit	Property	The unit of measure for the value.

7.4 Property implementation requirements

The client and the implementer should understand that, regardless of the schema declaration, an implementer may choose to not implement a property. If not implemented, a representation of the property will not be present in a reply. This should not be confused with a response that indicates that a property has been implemented, but has no value (i.e. *propertyName* = *null*).

There are several factors that could affect the implementation choice. Implementation requirements can be defined in many documents. At a minimum, a developer should review, in order: 1. the Redfish specification, 2. this document, and 3. associated profile specifications.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

7.5 Schema repository

The primary online source for the Swordfish schema shall be co-located on the DMTF schema site with the Redfish schema: http://redfish.dmtf.org/schemas/swordfish Developers may also download the schema as part of the Swordfish bundle from snia.org (refer to snia.org/swordfish for pointers to the bundle locations).

Implementations should refer either to the versions available on the dmtf.org site or to locally provided instances of the schema.

7.6 Referencing other schemas

Swordfish directly references many Redfish schemas when functionality is already defined and can be leveraged. Other Redfish schema may be added by inference or directly to implementations. Examples are available in the Swordfish mockups.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

8 Implementation requirements

8.1 Security

This document generally adheres to the security requirements defined in the Redfish Specification. It extends the Redfish security model in one important way:

- Swordfish implementations shall implement TLS as per the guidance in ISO/IEC 20648 and the TLS Specification for Storage Systems.

8.2 General constraints

8.2.1 Redfish elements

The Swordfish service interface extends the Redfish service interface. As such, a Swordfish service is a Redfish service and all required elements of the Redfish model shall be present in a Swordfish model.

Swordfish functionality shall not conflict with any previously defined Redfish functionality but it may add to or extend it, and it may add additional constraints on Redfish functionality.

Additionally, any functionality desired in a Swordfish implementation that is specified in Redfish shall follow the requirements as specified in the Redfish specification.

8.2.2 Storage Events

8.2.2.1 Overview

A Swordfish implementation should implement an event service. Redfish defines the Event Service framework, client subscription model, event delivery mechanism, as well as standard message registries. Swordfish extends the standard message registries to provide additional message registries that correspond to Swordfish-specific services and properties.

The Redfish event service publishes a list of event types supported, and maintains a list of clients that have subscribed. Each subscription maps clients, subscribed events, and the resources that generate them.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

8.2.2.2 Message Registry Selection and Management

Swordfish constrains the existing event model to provide a more consistent handling of event notifications and the related messages, in order to assure that client systems can easily and consistently parse and respond to system-level events.

8.2.2.3 Required Usage

- The Resource Event Message Registry defines the underlying messaging model, and shall be used to map messages to resources for storage implementations.
- The Redfish Base Message Registry shall be used to support HTTP connection/error/protocol issues, and general errors.
- The Swordfish Message Registry shall be used as a supplement for the resource event message registry.
- If the Swordfish service implements Redfish tasks (i.e., long-running operations), the implementation shall use the messages defined in the Task Event Message Registry to report status.

8.2.2.4 Recommended Usage

- Standard Messages should be used, wherever possible.
- OEM messages should be avoided. Suggestions for clarification or expansion of the existing registries are encouraged. (submissions should be sent to the SNIA Feedback Portal)

8.3 Discovering Swordfish resources

Each Swordfish implementation supports the following well-known URLs, as defined in Redfish. Specifically:

- `/Redfish`, which contains one or more version properties for the integrated Swordfish and Redfish implementation, starting with v1.
- `/Redfish/v1`, which addresses a `ServiceRoot` instance, which defines the Redfish default principal starting information for version 1 implementation of an integrated Redfish and Swordfish service. A GET operation to it shall retrieve the value of an instance of a `ServiceRoot` `EntityType` as defined in the ServiceRoot_v1.xml file.
- `/Redfish/v1/odata`, which addresses a `ServiceContainer` instance, which defines OData conformant principal starting information for the same version 1 implementation of an integrated Redfish and Swordfish service. A GET operation shall retrieve the value of an instance of a `ServiceContainer` `EntityContainer` as defined in the ServiceRoot_v1.xml file.

> Note: Since the `ServiceContainer` is required to return an `@odata.context` value of `/redfish/v1`, all other elements accessed via it will be the same elements found via the `ServiceRoot`.

> Note: A Swordfish service is a Redfish service with extensions to support storage management. No additional service entry-points are necessary.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Both the `ServiceRoot` and `ServiceContainer` contain a resource collection named `Systems` that lists `ComputerSystem` instances. A `ComputerSystem` instance that supports Swordfish defined services will have a value of "StorageServer" in an entry of its `HostingRoles` property.

The `ServiceContainer` additionally has a Service attribute that references the `ServiceRoot` resource.

Regardless of starting point, the property values of the `ServiceRoot` instance enable navigation to all other resources exposed by the Swordfish service.

8.4 ClassOfService requirements

Each `ClassOfService` shall include at least one line of service. The providing server shall assure that the line of service values of a `ClassOfService` collectively represent a supported choice of service.

8.5 StorageSystems requirements

For Hosted Service Configurations, this property of the `ServiceRoot` references a collection of `ComputerSystem` resources that each support Swordfish functionality. Each `ComputerSystem` included in the `StorageSystems` entry in the `ServiceRoot` shall have:

- an entry in its `HostingRoles` property with the value of `StorageServer`
- at least one entry in its `StorageServices.Members` property.

For Integrated Service Configurations, the StorageSystems concept is realized through the `StorageController` resource. Each `StorageController` instantiated as a Swordfish StorageSystem shall have:

- at least one entry in its `StorageController.Links` property `StorageServices` collection identifying related `StorageServices`

8.6 Entity Sets

The Swordfish model does not currently expose any explicitly defined entity sets. OData specifies that an entity set is defined for each NavigationProperty that is defined as a collection and that has the ContainsTarget attribute set to true. In all other cases, Swordfish assumes that an entity set is defined globally within the implementation for each entity type. This is effectively the same as if the entity sets were explicitly defined in the ServiceRoot entity container.

8.7 Addressing entities within a collection

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

An instance (entity) of an EntityType is uniquely identified within its entity set by its key. The URI for the reference may specify the key using one of two general strategies

1. OData recommends specifying the key value within parenthesis following the path segment that identifies the referencing entity set. (See clause "Canonical URL" in in OData)
2. Redfish common practice is to use an alternative form that adds a path segment having the value of the key following the path segment that identifies the referencing collection. (See clause "Alternate Key-as-Segment Syntax" in OData.)

A Swordfish implementation shall support both strategies.

8.8 Addressing members of a ResourceCollection

Redfish specifies that subclasses of ResourceCollection shall include a Members collection property (See clause "Collection resource response" in DSP0266)

Redfish allows a POST request to a ResourceCollection to be equivalent to the same POST request to the Members property of that ResourceCollection. For a particular ResourceCollection, if a Swordfish implementation supports either form, it shall support both.

It is common practice in Redfish to also eliminate the Members property from any request URI that navigates through a type hierarchy that includes a Member within a ResourceCollection. Care should be taken when defining and using a ResourceCollection subclass to not introduce ambiguities when an explicit reference to a Members property is dropped from a request URI.

8.9 HTTP status codes

8.9.1 Overview

Status codes are generally defined as part of the general HTTP protocol definition. In addition, the Redfish specification calls out general usage for HTTP status codes. This section provides additional usage guidance and constraints for Swordfish implementations.

In some instances, Redfish and Swordfish expand the standard use of HTTP status codes by associating additional system status with specific status codes. In addition, error response data may be included via standardized message registry entries. The specific messaging requirements will be defined in the following sections.

In cases where Swordfish adds additional constraints or expands on the Redfish handling of a given issue, this document will include both a clause reference (relative to the 1.7.0 version of the Redfish specification), and a small wording extract for additional context. For example:

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Swordfish refines the requirements in **x.y.z** of the *Redfish Specification*: `Redfish has no constraint on external storage functionality` to require that all references to external storage functionality shall be compliant with the current release of Swordfish.

8.9.2 Create

If a request to create a resource can be completed successfully without additional time, the Redfish service shall return a status code of 201, and the body of the response shall contain the JSON representation of the newly created resource.

If the create resource request has been accepted, but no information about the resource can be returned at this point, the Redfish service shall return a status code of 204. The payload of the response shall be empty, but the Location header shall contain the resource URI. The client will be required to poll the appropriate resource to determine both when and if the operation is complete.

Swordfish refines the requirements in clauses **7.5.1** and **12.2** of the *Redfish Specification*.

If a request to create a resource cannot be completed without additional time, the implementation shall:

- Populate an initial object. It shall contain, at a minimum, a valid URI, required properties (e.g., ID, name), and Status.State;
- Set Status.State of the partially populated resource to “Creating”;
- Return the appropriate status code, based on the following guidance:
  - If a Task Service has been implemented, the Redfish service shall return a status code of 202, with the Location header set to the URI of the Task Monitor. Once the provider has returned a Task Monitor to the client, the Client can then query the provided task URI to track the task completion status. Upon task completion, a GET against the task monitor may return a status code of 201, and the body of the message shall contain the created resource, provided the task monitor URI remains valid . Refer to the Redfish Task Manager documentation for the lifecycle of the task monitor URI.
  - If a Task Service has not been implemented, the Redfish service shall return a status code of 201, and the body of the response shall contain the URI of the skeletal resource created as part of accepting the request. The client will be required to poll the URI provided to determine when the operation is complete.
- Update Status.State for the object, once the create operation completes.

8.9.3 Update, Replace, Delete

If a request to modify or delete a resource can be completed without additional time, the Redfish service shall return a status code of 200, and the body of the response shall contain the JSON representation of the modified (or deleted) resource.

If the resource modification or deletion request has been accepted, but no information about the resource can be

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

returned at this point, the Redfish service shall return a status code of 204. The payload of the response shall be empty. The client will be required to poll the appropriate collection to determine both when and if the operation is complete.

If a request to modify a resource cannot be completed without additional time, the implementation shall:

- Set Status.State of the partially populated resource to “Updating” or “Deleting”, as appropriate;
- Return the appropriate status code, based on the following guidance:
- If a Task Service has been implemented, the Redfish service shall return a status code of 202, with the Location header set to the URI of the Task Monitor. Once the provider has returned a Task Monitor to the client, the Client can then query the provided task URI to track the task completion status. Upon task completion, a GET against the task monitor may return a status code of 201, and the body of the message shall contain the created resource, provided the task monitor URI remains valid . Refer to the Redfish Task Manager documentation for the lifecycle of the task monitor URI.
- If a Task Service has not been implemented, the Redfish service shall return a status code of 200, and the body of the response shall contain the URI of the skeletal resource created as part of accepting the request. The client will be required to poll the URI provided to determine when the operation is complete.
- For an update or replace request, the implementation shall update Status.State for the resource, once the modify operation completes.

8.9.4 Actions

Swordfish supports the approach to Actions in **5.6.3** of the *Redfish Specification*: `Actions are Redfish operations that do not easily map to RESTful interface semantics. These types of operations may not directly affect properties in the Redfish Resources.`

Swordfish refines the requirements in **7.10** of the *Redfish Specification*: `Services shall support the POST method to send actions to Resources.`

If a Task Service has been implemented, the Redfish service shall return a status code of 202, with the Location header set to the URI of the Task Monitor. Once the provider has returned a Task Monitor to the client, the Client can then query the provided task URI to track the task completion status. Once the task has completed successfully, a GET against the task monitor shall return the created object.

If a Task Service has not been implemented, the Redfish service shall return a status code of 200, and the body of the response shall contain the URI of the skeletal resource created as part of accepting the request. The client will be required to poll the URI provided to determine when the operation is complete. When processing ACTIONS, the handling of HTTP status codes is slightly different than that seen when processing CREATE or MODIFY requests. The HTTP status code is used to reflect the acceptance and formatting of the request. The outcome of any requested processing is reflected in the body of the returned message and its associated Error response structure. For example, a properly formatted request to execute a system reset may return an HTTP status code of 200 (OK), to reflect that the request has been received, was validly formatted, and has been accepted for processing, while the reset of the

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

system may not complete successfully. The Error response structure would contain further detail of the success of failure of the system reset. The implementation must check both the HTTP status code and the underlying Error response message structure to confirm the successful execution of the ACTION.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

9 Swordfish type definitions

9.1 Overview

The following sections define the schema and type definitions that make up a Swordfish implementation. Each data type or entity within the schema includes a description that defines its implementation requirements and their interaction

9.2 Common properties

9.2.1 Properties defined for all Redfish schemas

The properties summarized in Table 9 are included in every Redfish schema, and therefore may be encountered in any Response payload. They are documented here to avoid repetition in the Resource Guide tables for each schema.

Table 9: Common Properties

Property	Datatype	Attributes	Notes
@odata.context	string	read-only	The @odata.context property is a URL to a metadata document with a fragment describing the data (typically rooted at the top-level singleton or collection). Technically the metadata document only has to define, or reference, any of the types that it directly uses, and different payloads could reference different metadata documents. However, since the @odata.context provides a root URL for resolving relative references (such as @odata.id's), we return the canonical metadata document.
@odata.id	string	read-only	The @odata.id property is a string that indicates the unique identifier of a resource.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

Table 9: Common Properties, cont.

Property	Datatype	Attributes	Notes
@odata.type	string	read-only	The @odata.type property is a URL fragment that indicates the type of the resource.
Description	string	read-write	The Description property is used to convey a human-readable description of the resource.
Id	string	read-write	The Id property of a resource uniquely identifies the resource within the Resource Collection that contains it. The value of Id is unique within a Resource Collection.
Name	string	read-write	The Name property is used to convey a human-readable moniker for a resource. The type of the Name property is a string. The value of Name is NOT necessarily unique across resource instances within a Resource Collection.
Oem { }	object	read-write	This is the manufacturer/provider specific extension moniker used to divide the Oem object into sections. See the Resource schema for details on this property.

9.2.2 Links

The Links property represents the links associated with the resource, as defined by that resource's schema definition. All associated reference properties defined for a resource are nested under the Links property. All directly referenced (subordinate) properties defined for a resource can be found from the root of the resource.

9.2.3 Actions

The Actions property contains the actions supported by a resource.

9.2.4 OEM

The OEM property is used for OEM extensions.

9.2.5 RelatedItem

The RelatedItem property is represented as a set of links. The links point to a resource, or part of a resource, as defined by that resource's schema definition.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

This representation is not intended to be a strong linking methodology like other references. Instead it is used to show a relationship between elements or sub-elements in disparate parts of the service. For example, Fans may be in one area of the system and Processors in another area of the system. It could be that the relationship between the two is not obvious. The RelatedItem property can be used to show that one is related to the other. In this example, it might indicate that a specific fan is cooling a specific processor.

9.2.6 Status

9.2.6.1 Overview

The `Status` property is common to many Redfish schema. Its attributes are summarized in Table 10.

Table 10: Status property attributes

Health	string (enum)	read-only	This represents the health state of this resource in the absence of its dependent resources. See Health in Property Details, below, for the possible values of this property.
HealthRollup	string (enum)	read-only	This represents the overall health state from the view of this resource. See HealthRollup in Property Details, below, for the possible values of this property.
Oem { }	object	read-write	Oem extension object.
State	string (enum)	read-only	This indicates the known state of the resource, such as if it is enabled. See State in Property Details, below, for the possible values of this property.

9.2.6.2 Property details

Health:

This represents the health state of this resource in the absence of its dependent resources. Its possible values are summarized in Table 11.

Table 11: Health ENUM values, cont.

string	Description
Critical	A critical condition exists that requires immediate attention.
OK	Normal.
Warning	A condition exists that requires attention.

HealthRollup:

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

This represents the overall health state from the view of this resource. Its possible values are summarized in Table 12.

Table 12: HealthRollup ENUM values

string	Description
Critical	A critical condition exists that requires immediate attention.
OK	Normal.
Warning	A condition exists that requires attention.

State:

This indicates the known state of the resource, such as if it is enabled. Its possible values are summarized in Table 13.

Table 13: State ENUM values, cont.

string	Description
Absent	This function or resource is not present or not detected.
Disabled	This function or resource has been disabled.
Enabled	This function or resource has been enabled.
InTest	This function or resource is undergoing testing.
Quiesced	The element is enabled but only processes a restricted set of commands.
StandbyOffline	This function or resource is enabled, but awaiting an external action to activate it.
StandbySpare	This function or resource is part of a redundancy set and is awaiting a failover or other external action to activate it.
Starting	This function or resource is starting.
UnavailableOffline	This function or resource is present but cannot be used.
Updating	The element is updating and may be unavailable or degraded.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 5965:2021

9.2.7 Location

9.2.7.1 Location properties overview

The properties of a Location object are summarized in Table 14.

Table 14: Location Properties

Property	Type	Notes	
AltitudeMeters	number (m)	read-only (null)	The altitude of the resource in meters.
Info	string	read-only (null)	This indicates the location of the resource.
InfoFormat	string	read-only (null)	This represents the format of the Info property.
Latitude	number (deg)	read-only (null)	The latitude resource.
Longitude	number (deg)	read-only (null)	The longitude resource in degrees.
Oem { }	object	read-write	Oem extension object. See the Resource schema for details on this property.
PartLocation {	object	read-write	Postal address of the addressed resource.
LocationOrdinalValue	number	read-only (null)	The number that represents the location of the part. If LocationType is slot and this unit is in slot 2 then the LocationOrdinalValue will be 2.
LocationType	string (enum)	read-only	The type of location of the part, such as slot, bay, socket and slot. See LocationType in Property Details, below, for the possible values of this property.
Orientation	string (enum)	read-only	The orientation for the ordering of the slot enumeration used by the LocationOrdinalValue property. See Orientation in Property Details, below, for the possible values of this property.

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 5965:2021