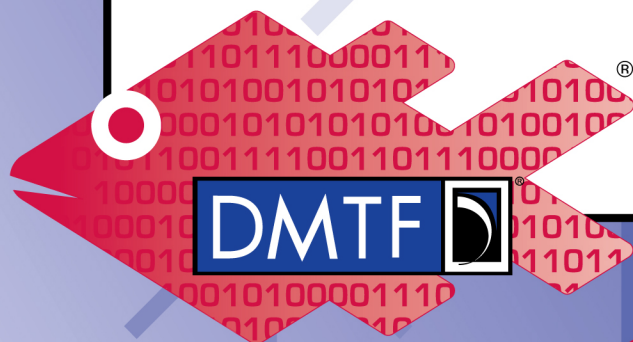# Redfish Eventing

Jeff Hilland
**HPE**

Agenda

- Resource Overview
- Eventing Model Map
- Examples
    - Event Service
    - How to subscribe
    - What Events look like

Resource Overview

- Event Service: A representation the event service inside of the Redfish Service
  - Contains Attributes of the service:
    - Such as status, retry information, event types for subscription and an action which allows a test event to be sent.
  - Contains a collection of Subscriptions
    - Clients post to this collection to subscribe to Events
- Event Subscriptions: A representation of the event destination in the client.
  - Contains the URL of the destination for events to be sent
  - Also contains the list of event types to be sent, a user supplied context and protocol type (only allowed to be "Redfish" currently)
- Events: The message sent on the subscription connection
  - The events right now are not in the mockups. That's because they don't exist in the service and are instead sent on the subscription connection.
- Message Registries: An array of messages and their attributes organized by MessageId .
  - Organized by MessageId, each entry has a description, the message the Id translates to, Severity, Number and type of arguments and a proposed resolution.
  - The arguments are basically substitution variables for the message so the client can full render it for the user.
  - Redfish Events, Errors and Log Records have a unified format.
  - Message Registries can be translated for localization and have the "Languages" property to indicate which translation of the message each particular registry represents.

Event Service Model Map

www.dmtf.org

Copyright 2018 DMTF
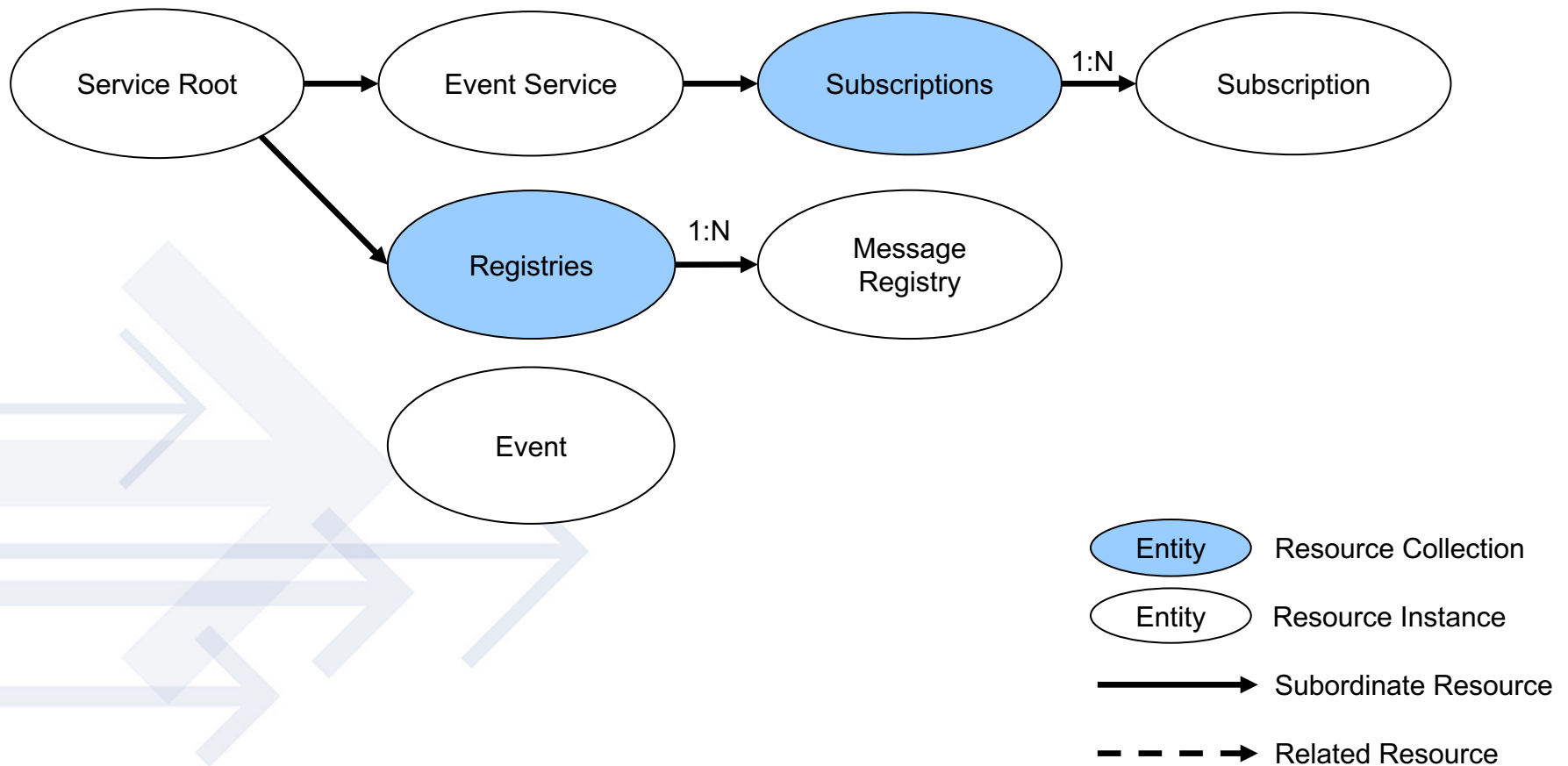
Event Service

```
{
    "@odata.context": "/redfish/v1/$metadata#EventService.EventService",
    "@odata.type": "#EventService.v1_0_0.EventService",
    "Id": "EventService",
    "Name": "Event Service",
    "Status": {
        "State": "Enabled",
        "Health": "OK"    },
    "ServiceEnabled": true,
    "DeliveryRetryAttempts": 3,
    "DeliveryRetryIntervalSeconds": 60,
    "EventTypesForSubscription": [
        "StatusChange",
        "ResourceUpdated",
        "ResourceAdded",
        "ResourceRemoved",
        "Alert"
    ],
    "Subscriptions": {"@odata.id": "/redfish/v1/EventService/Subscriptions"},
    ...
```

Retry

Event Types

Subscriptions

Copyright 2018 DMTF

EventSubscription

```
{
    "@odata.context": "/redfish/v1/$metadata#EventDestination.EventDestination",
    "@odata.id": "/redfish/v1/EventService/Subscriptions/1",
    "@odata.type": "#EventDestination.v1_0_0.EventDestination",
    "Id": "1",
    "Name": "EventSubscription 1",
    "Destination": http://www.dnsname.com/Destination1,
    "EventTypes": [
        "Alert"
    ],
    "Context": "ABCDEFGHJLKJ",
    "Protocol": "Redfish"
}
```

Destination URI

Event Types

User Supplied Context & Protocol Selection

Event

```
{
    "@odata.context": "/redfish/v1/$metadata#Event.Event",
    "@odata.id": "/redfish/v1/EventService/Events/1",
    "@odata.type": "#Event.v1_1_0.Event ",
    "Id": "1",
    "Name": "Event Array",
    "Context": "ABCDEFGH",
    "Events": [
        {
            "EventType": "Alert",
            "EventId": "ABC132489713478812346",
            "Severity": "Warning",
            "Message": "The LAN has been disconnected",
            "MessageId": "Alert.1.0.LanDisconnect",
            "MessageArgs": [
                "EthernetInterface 1",
                "/redfish/v1/Systems/1"
            ],
            "OriginOfCondition": {
                "@odata.id": "/redfish/v1/Systems/1/EthernetInterfaces/1"
            },
            "Context": "ABCDEFGH"
        }
    ]
}
```

Type and Grouping

Optional but unlikely

Registry Decode Info

Arguments for Message

Resource causing the message

Message Registry (example entry, not official)

```
{
    "@odata.type": "#MessageRegistry.v1_0_0.MessageRegistry",
    "Id": "Alert.1.0.0",
    "Name": "Base Message Registry",
    "Language": "en",
    "Description": "This registry is a sample Redfish alert message registry",
    "RegistryPrefix": "Alert",
    "RegistryVersion":  "1.0.0",                    Registry Prefix & Version
    "OwningEntity": "Contoso",
    "Messages": {
        "LanDisconnect": {                          MessageId
            "Description": "A LAN Disconnect on %1 was detected on system %2.",
            "Message": "A LAN Disconnect on %1 was detected on system %2.", Message for Display &
            "Severity": "Warning",                  Severity
            "NumberOfArgs": 2,                       Types of args in the msg
            "Resolution": "None"
        },
    ...
```

How a client finds the message in a registry a forms a message

1. Use the MessageId to find the registry, version and message
   - The event in the example had the MessageId "Alert.1.0.LanDisconnect"
     - The segment before the 1st period is the Registry Name (Registry Prefix): "Alert"
     - The segment between the 1st and 2nd period is the major version: "1"
     - The segment between the 2nd and 3rd period is the minor version: "0"
     - The segment after the 3rd period is the Message Identifier in the Registry: "LanDisconnect".

2. Use that information to find the registry and the message in it.
   - The ServiceRoot has a link to the MessageRegistry collection.
     - Find the member of the collection that has the RegistryPrefix matching the first segment.
     - Find the Registry with the RegistryVersion matching the 2nd and 3rd segments
     - Find the element in the Messages array of that registry that matches the last segment.
     - Note that the DMTF publishes these at http://redfish.dmtf.org\registries using the file named <RegistryPrefix>.<version>.json (ex: Base.1.0.0.json)

3. Populate the message for the end user
   - Take the Message property value and start substituting the message args from the event itself.
     - The Message Registry entry has the number of args.
     - Replace %1 in the Message string with the first arg, %2 with the second, and keep up this pattern until all args have been substituted.

Example with all other properties removed

Event:

```
{ "Events": [ {
          "MessageId": "Alert.1.0.LanDisconnect",
          "MessageArgs": [ "EthernetInterface 1", "/redfish/v1/Systems/1"]
} ] }
```

Message Registry & Entry:

```
{
    "RegistryPrefix": "Alert",
    "RegistryVersion":  "1.0.0",
    "Messages": {
        "LanDisconnect": {
            "Message": "A LAN Disconnect on %1 was detected on system %2.",
            "NumberOfArgs": 2,
} } }
```

Example with all other properties removed

Event:

```
{ "Events": [ {
          "MessageId": "Alert.1.0.LanDisconnect",
          "MessageArgs": [ "EthernetInterface 1", "/redfish/v1/Systems/1"]
} ] }
```

Message Registry & Entry:

```
{
    "RegistryPrefix": "Alert",
    "RegistryVersion":  "1.0.0",
    "Messages": {
        "LanDisconnect": {
            "Message": "A LAN Disconnect on %1 was detected on system %2.",
            "NumberOfArgs": 2,
} } }
```

Note that this <u>Message</u> is in English but could be in any language and %1, %2, etc. are reserved values for substitution. The language of the registry is indicated in the "Language" property as defined in the MessageRegistry schema and there may be multiple translations of the same registry.

Example with all other properties removed

Event:
```
{ "Events": [ {
          "MessageId": "Alert.1.0.LanDisconnect",
          "MessageArgs": [ "EthernetInterface 1", "/redfish/v1/Systems/1"]
} ] }
```

Message Registry & Entry:
```
{
    "RegistryPrefix": "Alert",
    "RegistryVersion":  "1.0.0",
    "Messages": {
        "LanDisconnect": {
            "Message": "A LAN Disconnect on %1 was detected on system %2.",
            "NumberOfArgs": 2,
} } }
```

Constructed Message for User:
```
          "Message": "A LAN Disconnect on %1 was detected on system %2.",
```

Example with all other properties removed

Event:

```
{ "Events": [ {
          "MessageId": "Alert.1.0.LanDisconnect",
          "MessageArgs": [ "EthernetInterface 1", "/redfish/v1/Systems/1"]
} ] }
```

Message Registry & Entry:

```
{
    "RegistryPrefix": "Alert",
    "RegistryVersion":  "1.0.0",
    "Messages": {
        "LanDisconnect": {
            "Message": "A LAN Disconnect on %1 was detected on system %2.",
            "NumberOfArgs": 2,
} } }
```
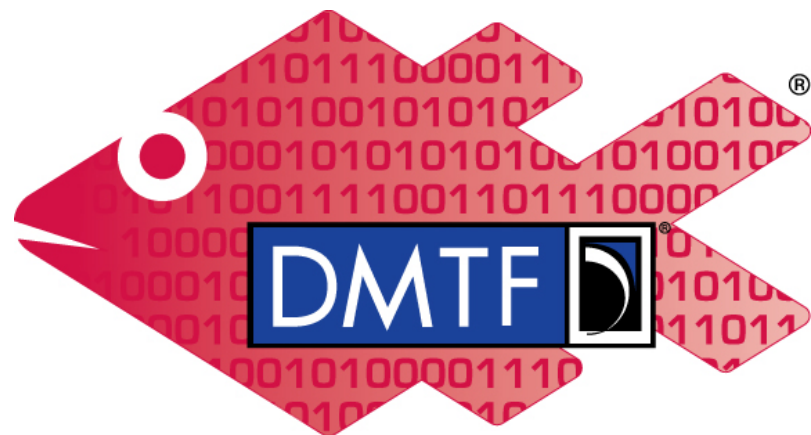
Constructed Message for User:

```
          "Message": "A LAN Disconnect on EthernetInterface 1 was detected
          on system /redfish/v1/Systems/1.",
```

Copyright 2018 DMTF

Thank you for watching!

- Redfish Standards
  - Schemas, Specs, Mockups, White Papers, FAQ, Educational Material & more
  - http://www.dmtf.org/standards/redfish
- Redfish Developer Hub
  - Redfish Interactive Explorer, Hosted Schema at Namespace & other links
  - http://redfish.dmtf.org
- SPMF (WG that defines Redfish)
  - Companies involved, Upcoming Schedules & Future work, Charter, Information on joining.
  - http://www.dmtf.org/standards/spmf