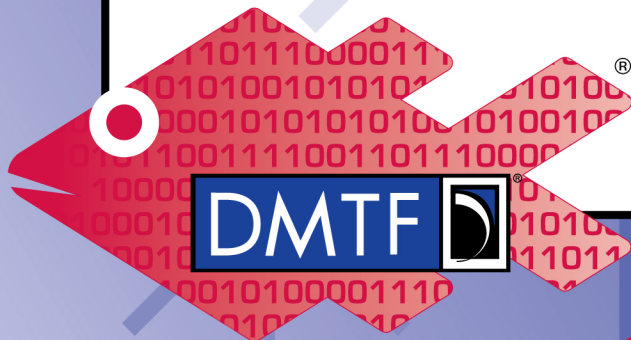




Redfish Events & Message Changes in 2018.2

Jeff Hilland
HPE



Redfish



Agenda

Note: This video assumes you have seen the first video on Redfish Events & Messages

- Changes in 2018.2
- Example of Clearing Logic Usage



Changes around Redfish Messages

- **Subscription Model**
 - Deprecated EventType in favor of RegistryPrefixes, ResourceTypes. Clients can now subscribe to specific Registries or specific Resource Types instead of loosely defined classes.
 - Added SubordinateResources which can be used in conjunction with ResourceOrigin
- **Grouping Mechanism**
 - Added EventGroupId property for clients to tell messages have the same root cause
- **Telemetry Support**
 - EventFormatType can now be used to get events that either conform to the Event or MetricReport schema
- **SSE Filtering**
 - Added language to indicate use of \$filter for SSE event subscriptions
- **New Registry Properties**
 - Added properties to support clearing logic so clients can tell which messages are indicating a previous situation has been cleared up.
- **New Message Registries**
 - Added Task and Resource Message Registries.
 - Expect more in the future that are more device/service specific.
- **OEM Registries**
 - Clarified the spec on rules allowing OEMs to extend messages in the standard registries.



Subscription Model Changes

- Modified EventService and EventSubscription to support:
 - EventType – deprecated.
 - RegistryPrefixes
 - The Event Service would have the Prefix of the Message Registries that the Client can subscribe to.
 - This value does not include version. That's in the /Registries files themselves that you can discovery as before.
 - If RegistryPrefixes is empty on subscription, the Client is subscribing to all Message Registries
 - ResourceTypes
 - Indicates the Resource Types (Schema names / classes) that the Client wants to receive messages about.
 - If ResourceType is empty on subscription, the Client is subscribing to receive events regardless of ResourceType.
 - SubordinateResources
 - Boolean that indicates if the Client subscribed to a specific Resource using OriginResources, that the client is also requesting events for any and all subordinate resources regardless of odata.type or depth.



Grouping Mechanism

- EventGroupId was added to Message
 - Is used by the Redfish Service to group messages together
 - Indicates that they came from the same event.
 - Example is that a cable unplugged could result in several messages.
 - Perhaps a resource changed message,
 - a status changed message
 - and an alert LAN.1.0.Disconnect message, for instance.
 - By setting them all to the same EventGroupId, the service is telling the client that all of these events arose from the same root cause.



Enable Telemetry Events

- Add a property called:
 - EventFormatType on EventDestination
 - EventFormatTypes on EventService
- This is an enum that is either “Event” or “MetricReport”
 - Describes the ResourceType (Schema) of the payload (JSON Body) sent to the Event Destination.
 - If it is Event, then it’s a normal event.
 - If it is MetricReport, then the destination gets MetricReports when the associated Trigger condition is met
- Also allow query strings on the ServerSentEventURI for SSE
 - Optional to support
 - Allows for MetricReport from the same MetricReportDefinition to be sent.
 - Service has a SSEFilterPropertiesSupported property to indicate support



Modify Message Schema

- Add “ClearingLogic” object to indicate how an event is cleared.
- This would be in any message that is an “assert”.
 - Add “ClearsIf” to Message indicating on what conditions it is cleared
 - Right now the only property we can foresee this for is “SameOriginOfCondition” but maybe parent or related origins could clear it.
 - Add “ClearsMessage” array of MessageIds indicating which message this message clears.
 - Example would be LanDisconnect clears on LanConnect if SameOriginOfCondition
 - Add “ClearsAll” Boolean indicating this clears all other alerts.
 - This is really for reset/reboot messages
 - This should be at the system level but a NIC could use ClearsAll with ClearsIf “SameOriginOfCondition”.
 - This should really be rare but if the device wakes up, it needs to clear previous problems for the client.



Clearing Logic Example

- Here are two messages in a Registry:
 - Note that Message, NumberOfArgs, ParamTypes, Resolution and Severity have been removed for simplicity

```
"ResourceErrorsDetected": { }  
"ResourceErrorsCorrected": {  
  "ClearingLogic": {  
    "ClearsIf": "SameOriginOfCondition",  
    "ClearsMessage": [  
      "ResourceErrorsDetected"  
    ]  
  }  
}
```
- This tells the Client that:
 - if it receives a ResourceErrorsCorrected message
 - and it had previously received a ResourceErrorsDetected message
 - then the condition of that resource causing that error has been cleared
 - if the OriginOfCondition for both messages has the same value.



Redfish has two new standard Message Registries

- **Resource**

- Submitted to DMTF from SNIA and published with modifications to match the 2018.2 changes
- Resource Life Cycle:
 - ResourceCreated, ResourceRemoved, ResourceChanged, URIForResourceChanged
- Error:
 - ResourceErrorsDetected, ResourceErrorsCorrected
 - ResourceErrorThresholdExceeded, ResourceErrorThresholdCleared
- Warning:
 - ResourceWarningThresholdExceeded, ResourceWarningThresholdCleared
- Status:
 - ResourceStatusChangedOK, ResourceStatusChangedWarning, ResourceStatusChangedCritical
- Self Test:
 - ResourceSelfTestFailed, ResourceSelfTestCompleted
- License:
 - LicenseExpired, LicenseChanged, LicenseAdded
- Misc:
 - ResourceVersionIncompatible

- **Task**

- Submitted to DMTF from SNIA and published with modifications to match the 2018.2 changes
- TaskStarted, TaskCompletedOK, TaskCompletedWarning, TaskAborted, TaskCancelled, TaskRemoved, TaskPaused, TaskResumed, TaskProgressChanged,



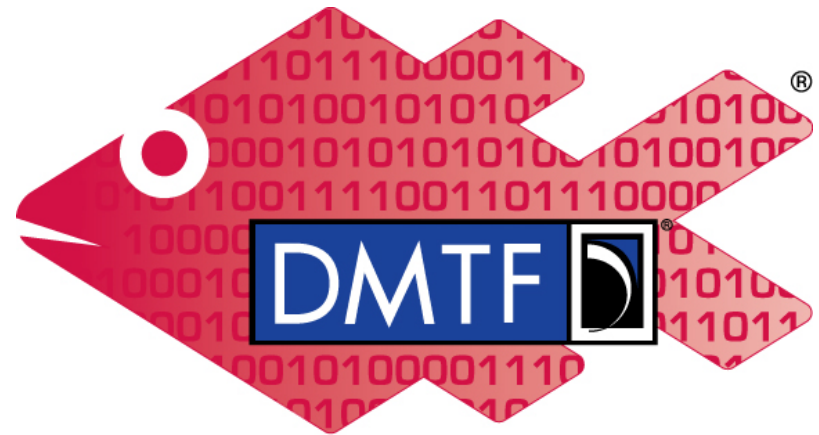
Extending messages

- How does OEM extend messages?
 - Services can have an OEM section inside of each entry in the Events array
- OEMs cannot use messageArgs beyond the last one in the registry
 - This is what the OEM section inside of each Event is for.
- Can an OEM use a substitute registry?
 - Services can have an OEM section in side of each standard message in a message registry
 - They cannot change any of the text in a standard message.
- OEMs can always include the Message property in each entry in the Events array.



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>
- Redfish Forum
(DMTF WG that defines Redfish)
 - Companies involved, Upcoming Schedules & Future work, Charter, Information on joining.
 - <http://www.dmtf.org/standards/spmf>



Redfish