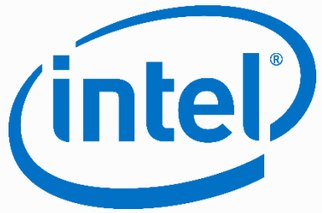
presented by



UEFI Manageability and REST Services

Fall 2017 UEFI Seminar and Plugfest
October 30 – November 3, 2017
Presented by Abner Chang (HPE), Ting Ye (Intel)

Agenda

- Introduction
- UEFI Manageability
- Real Use Case: UEFI iSCSI
- REST in UEFI
- Insufficient Capabilities in ‘EFI REST’ and Proposals for Enhancement
- Summary & Call to Action

Introduction

Introduction

- Today's platforms have amazing capabilities, but they are complex to configure
  - Firmware configuration model is still focused on "Press <DEL> to Enter Setup"
  - Remote management isn't consistent
- UEFI has abundant firmware manageability interfaces, and is adopting REST/Redfish, but there are still gaps in the standard

Goals

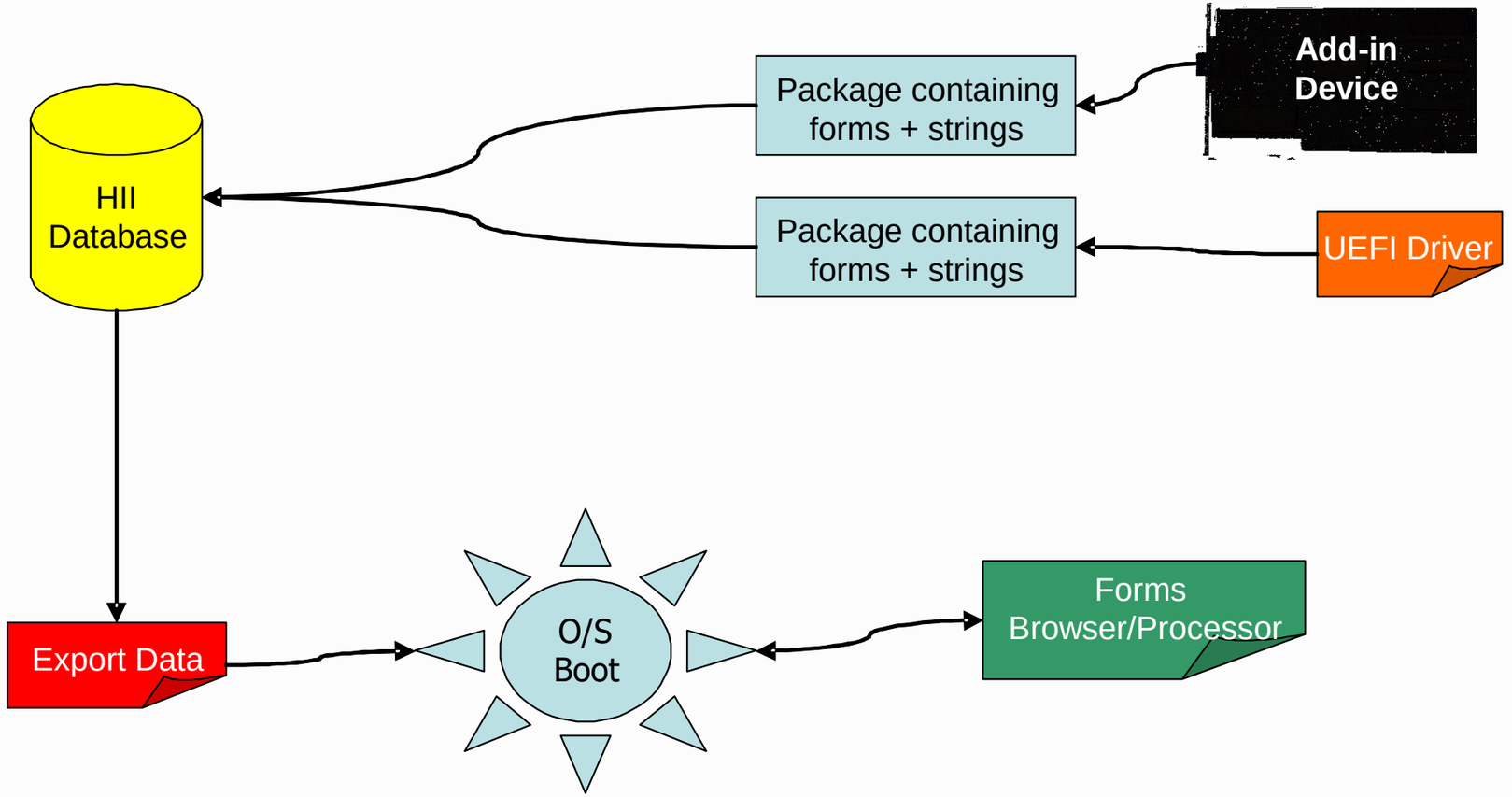
- Overview of UEFI manageability interfaces & Human Interface Infrastructure (HII)
- Examples of handling configuration data
- Overview of EFI REST protocol
- Proposals to cover gaps in REST services
- Background of REST service provider and UEFI REST Client

UEFI Manageability

UEFI Manageability

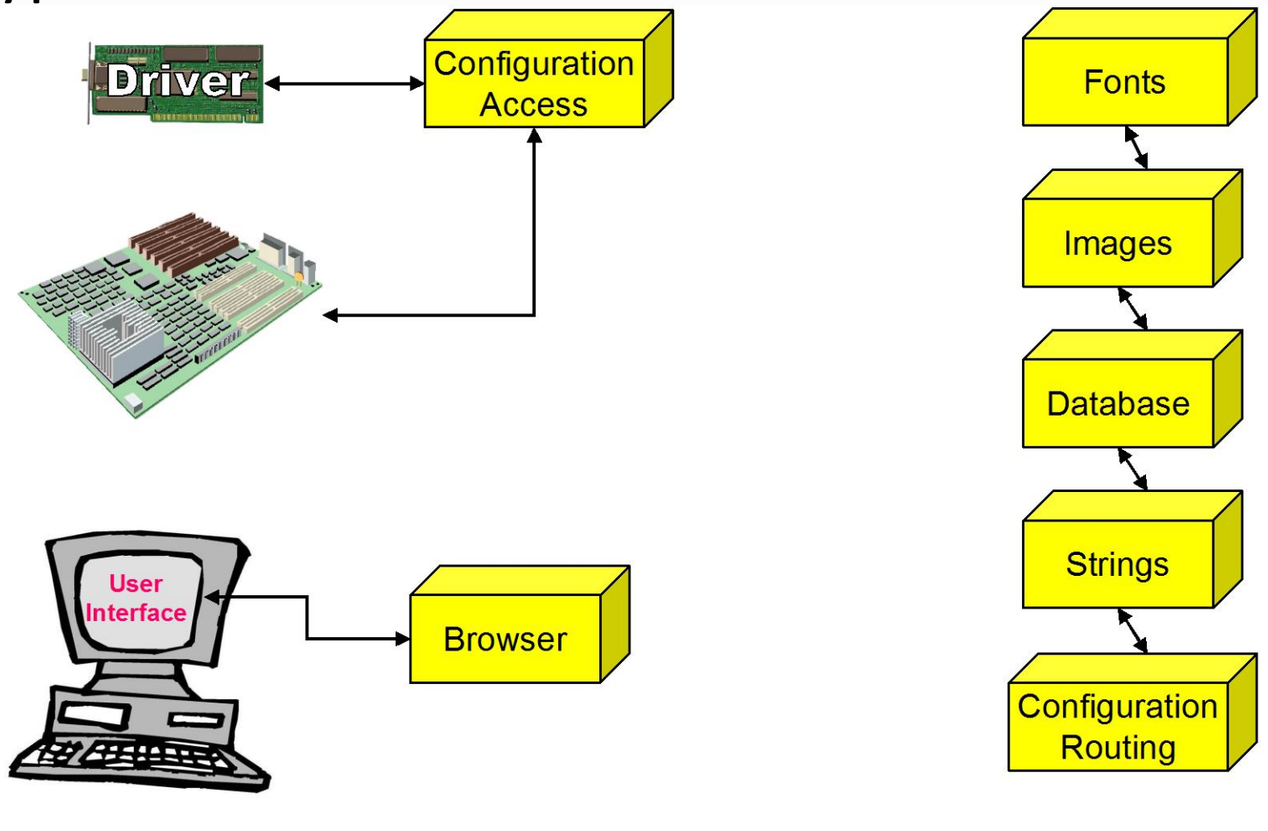
- Human Interface Infrastructure
  - HII Protocols
  - x-UEFI language
- Keyword Handler Protocol
- Configuration Namespace

Human Interface Infrastructure

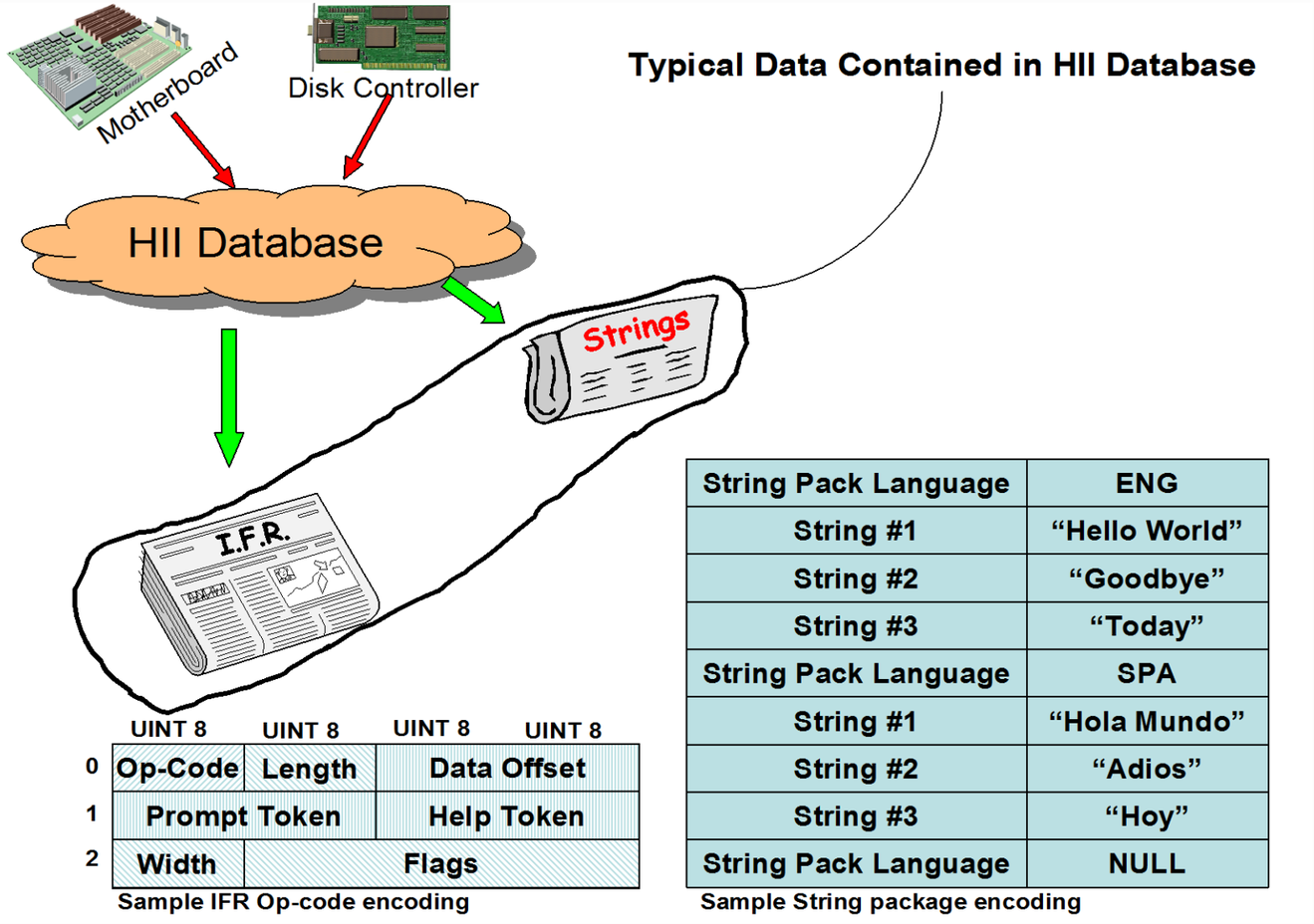


HII Protocols

- Content Registration:
  - HII Font, Font Ex, Font Glyph Generator
  - HII Image, Image Ex
    Image Decoder
  - HII String
  - HII Database
- Browser Protocol:
  - Config Access / Routing
  - Form Browser
  - HII Popup



Multiple Language Support



	UINT 8	UINT 8	UINT 8	UINT 8
0	Op-Code	Length	Data Offset	
1	Prompt Token		Help Token	
2	Width	Flags		

Sample IFR Op-code encoding

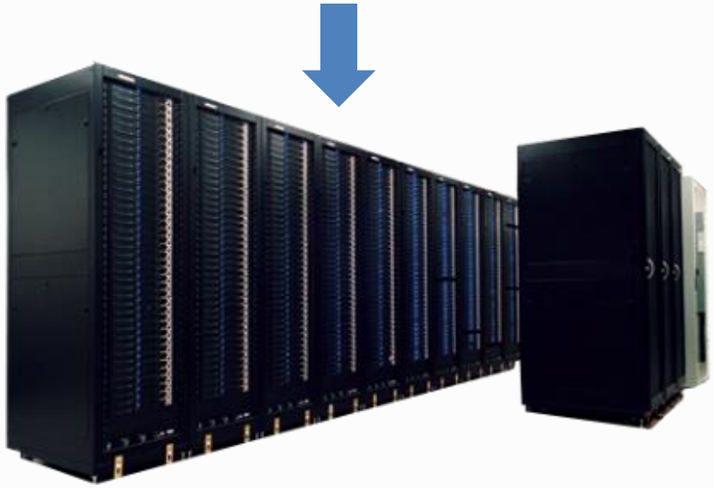
String Pack Language	ENG
String #1	“Hello World”
String #2	“Goodbye”
String #3	“Today”
String Pack Language	SPA
String #1	“Hola Mundo”
String #2	“Adios”
String #3	“Hoy”
String Pack Language	NULL

Sample String package encoding

x-UEFI: a “UEFI” Language

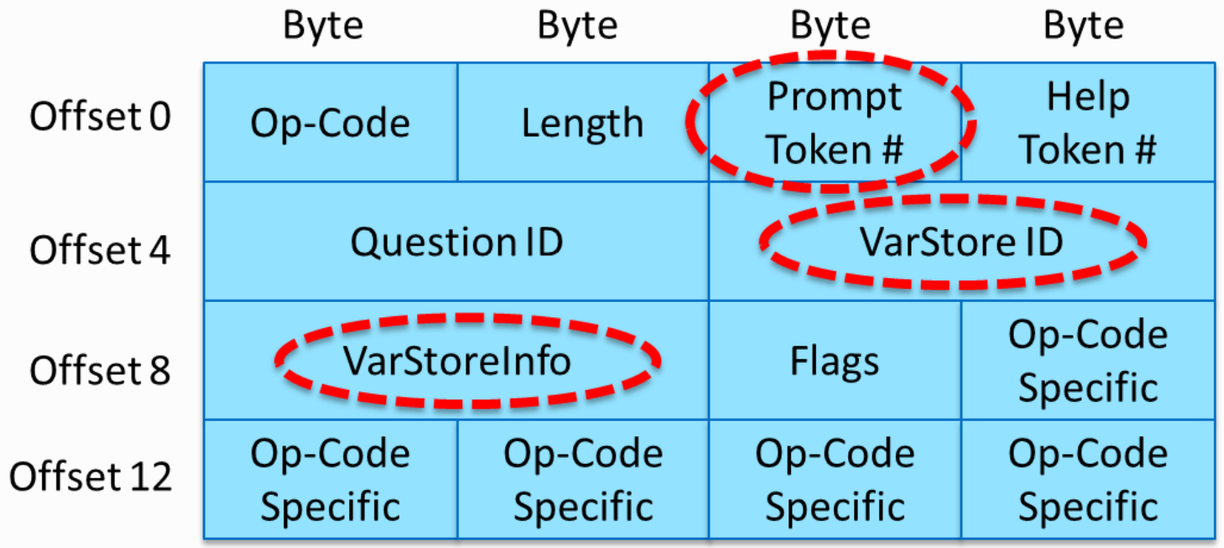
- “machine-to-machine” language

String Pack Language	ENG
Token #1	“BIOS Vendor”
String Pack Language	SPA
Token #1	“Vendedor de BIOS”
String Pack Language	x-UEFI
Token #1	“BIOS_VENDOR_KEYWORD”
.	
.	
.	
String Pack Language	NULL



How To Use This Language

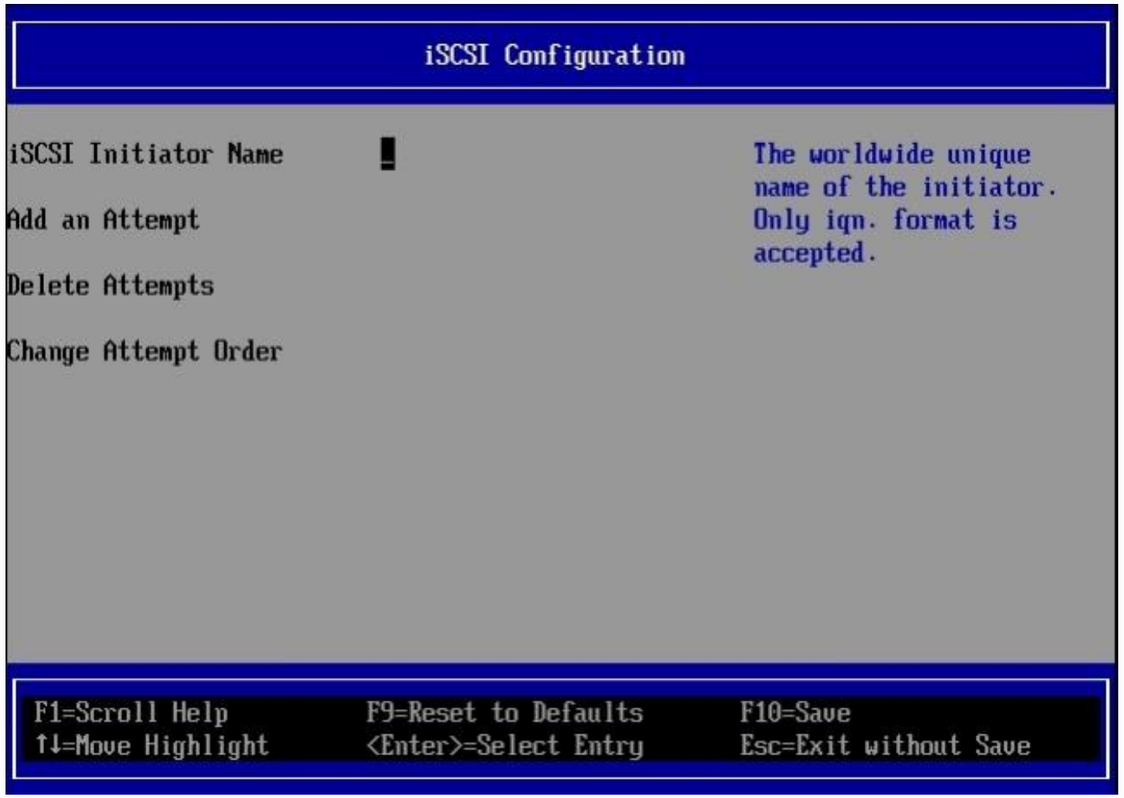
- Retrieve platform exported data
- Get String Token of x-UEFI keyword
- Find op-code using the string token

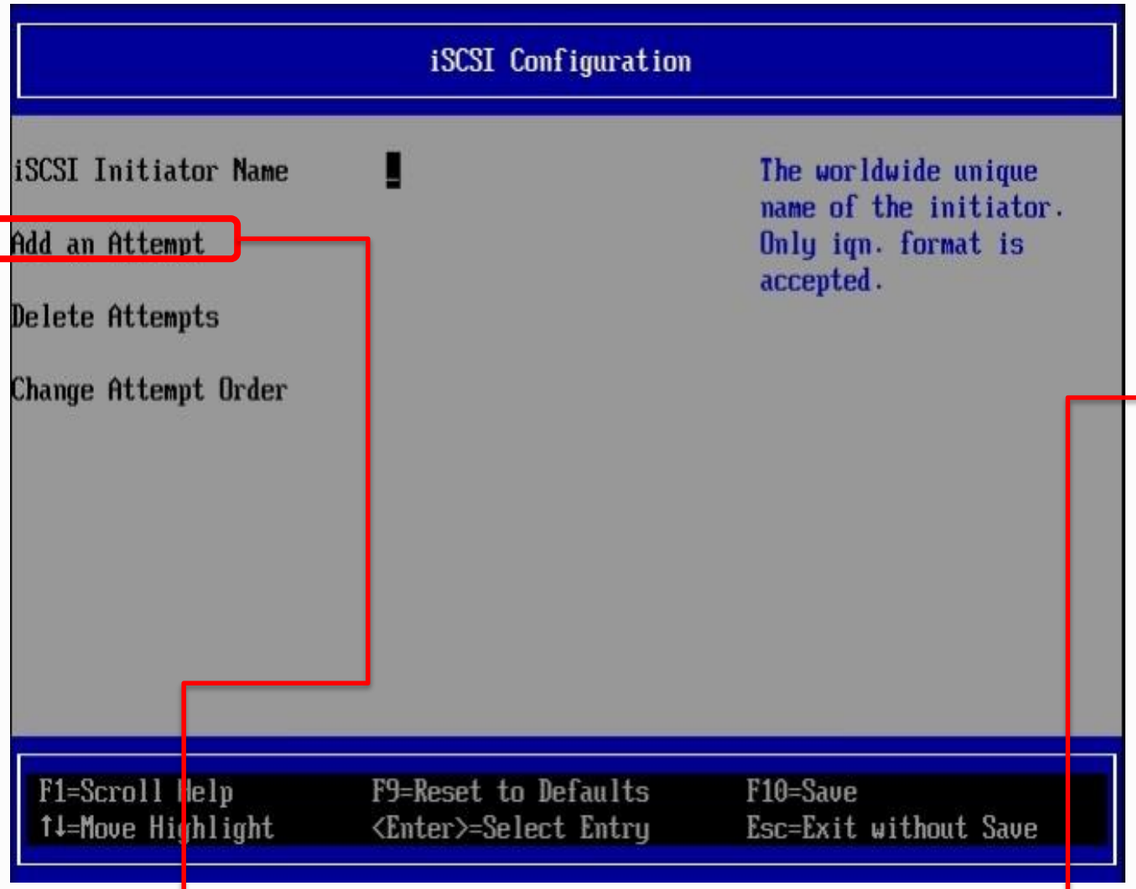


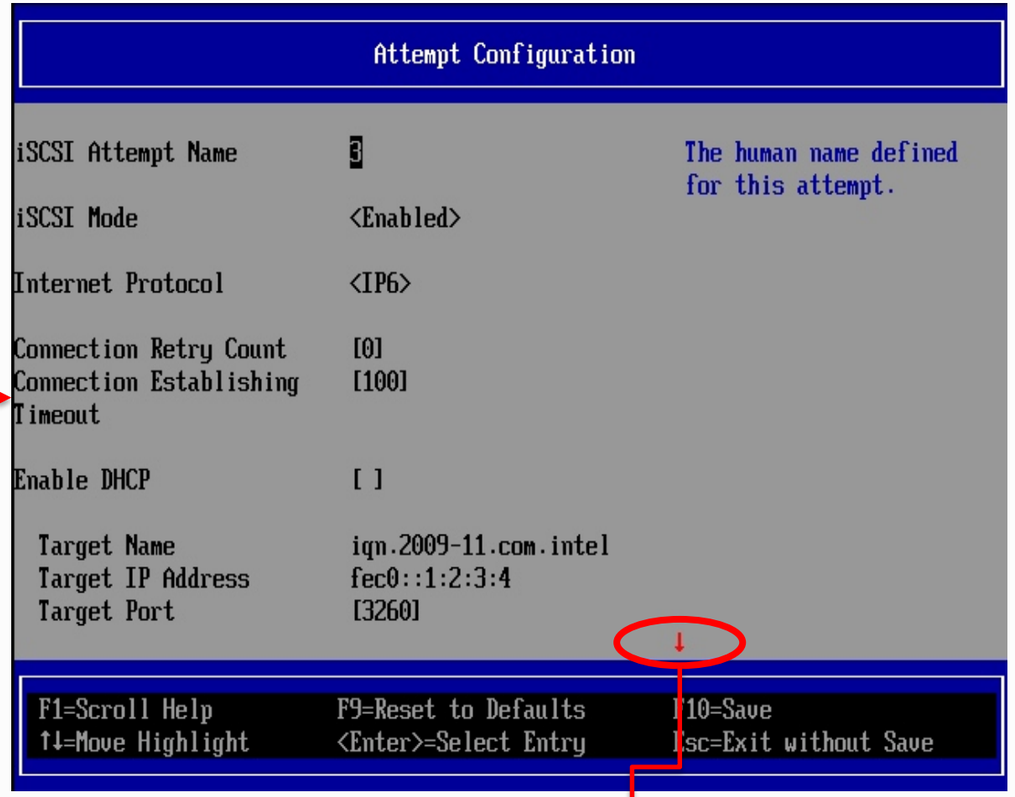
Real Use Case: UEFI iSCSI

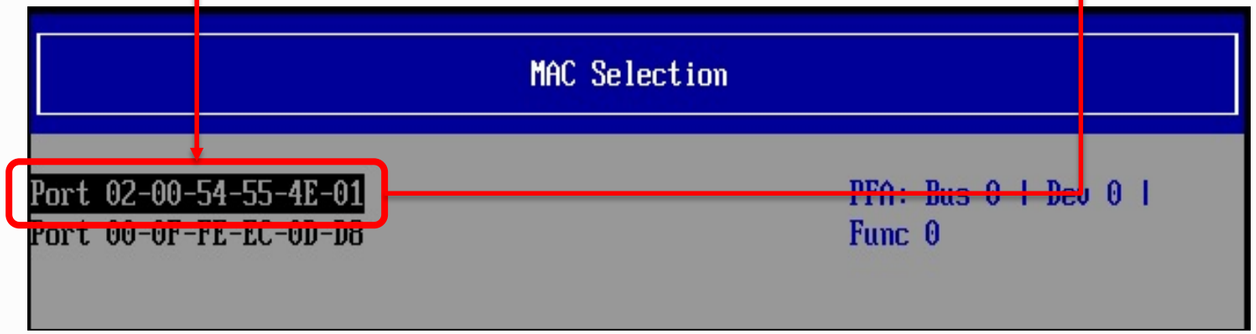
Real Use Case: UEFI iSCSI

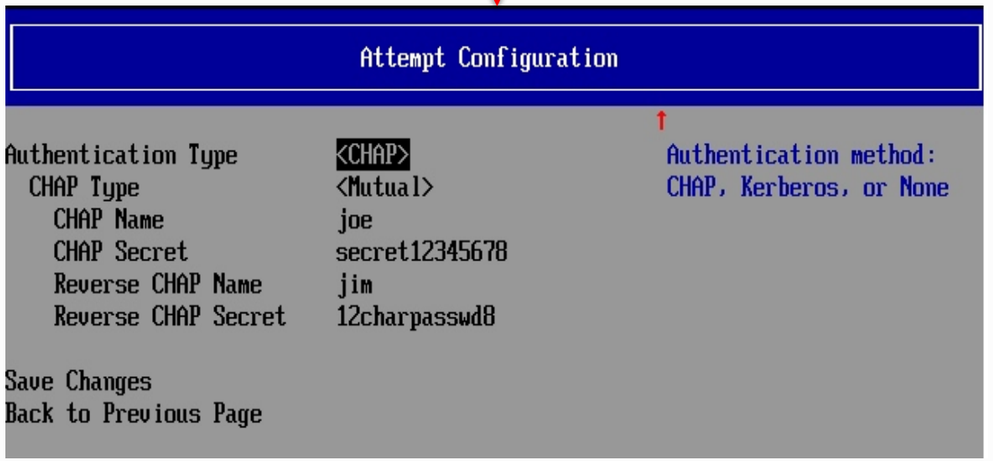
- iSCSI is a popular network boot target
- UEFI iSCSI defines a number of dynamic HII forms




iSCSI Configuration
iSCSI Initiator Name
Add an Attempt
Delete Attempts
Change Attempt Order
The worldwide unique name of the initiator. Only iqn. format is accepted.
F1=Scroll Help
↑↓=Move Highlight
F9=Reset to Defaults
<Enter>=Select Entry
F10=Save
Esc=Exit without Save


Attempt Configuration
iSCSI Attempt Name
3
iSCSI Mode
<Enabled>
Internet Protocol
<IP6>
Connection Retry Count
[0]
Connection Establishing Timeout
[100]
Enable DHCP
[ ]
Target Name
iqn.2009-11.com.intel
Target IP Address
fec0::1:2:3:4
Target Port
[3260]
The human name defined for this attempt.
F1=Scroll Help
↑↓=Move Highlight
F9=Reset to Defaults
<Enter>=Select Entry
F10=Save
Esc=Exit without Save


MAC Selection
Port 02-00-54-55-4E-01
Port 00-0F-FE-EC-0D-D8
PFA: Bus 0 | Dev 0 | Func 0


Attempt Configuration
Authentication Type
<CHAP>
CHAP Type
<Mutual>
CHAP Name
joe
CHAP Secret
secret12345678
Reverse CHAP Name
jim
Reverse CHAP Secret
12charpasswd8
Save Changes
Back to Previous Page
Authentication method: CHAP, Kerberos, or None

iSCSI x-UEFI keywords

- Registered to http://uefi.org/configfnamespace

iSCSIInitiatorName	ReadWrite	iSCSIMacAddr	ReadWrite
iSCSIAttemptName:#	ReadOnly	iSCSIAddAttempts	ReadWrite
iSCSIBootEnable:#	ReadWrite	iSCSIDeleteAttempts	ReadWrite
iSCSIConnectRetry:#	ReadWrite	iSCSIDisplayAttemptList	ReadOnly
……	……	……	……

x-UEFI Usage Example

```
iSCSIInitiatorName
```

VFR file

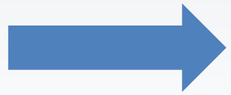
```
string  varid   = ISCSI_CONFIG_IFR_NVDATA.InitiatorName,
          prompt  = STRING_TOKEN(STR_ISCSI_CONFIG_INIT_NAME),
```

UNI file

```
#string STR_ISCSI_CONFIG_INIT_NAME    #language en-US "iSCSI Initiator Name"
#string STR_ISCSI_CONFIG_INIT_NAME    #language x-UEFI "iSCSIInitiatorName"
```

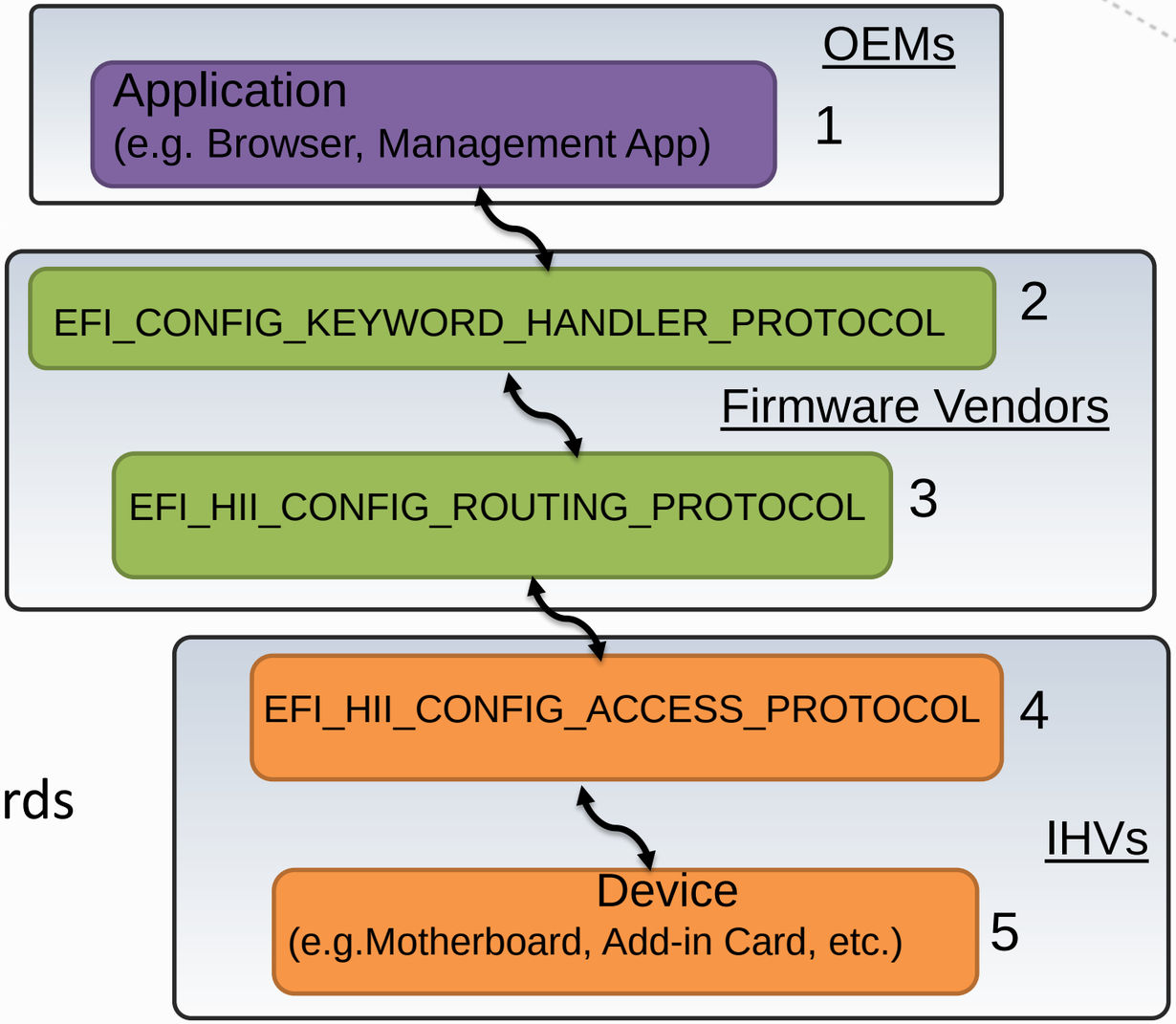
Script file

```
IScsiScript -i iqn.edkii.intel.com
```



General Guidelines

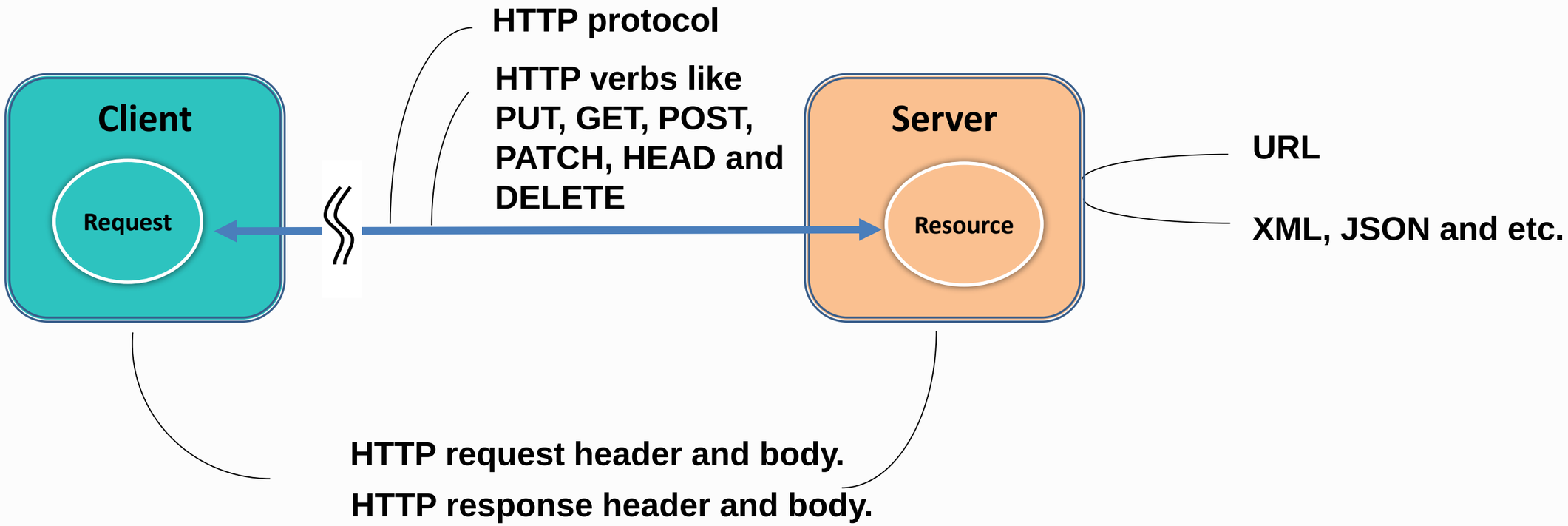
- OEMs …
  - Get keywords definition from http://uefi.org/configgamespace
  - Use KeywordHandler.GetData/ SetData
- Firmware vendors …
  - Get HII updates from the latest Intel® UEFI Development Kit
- IHVs …
  - Define and register x-UEFI keywords
  - Support keyword setting in ConfigAccess.RouteConfig



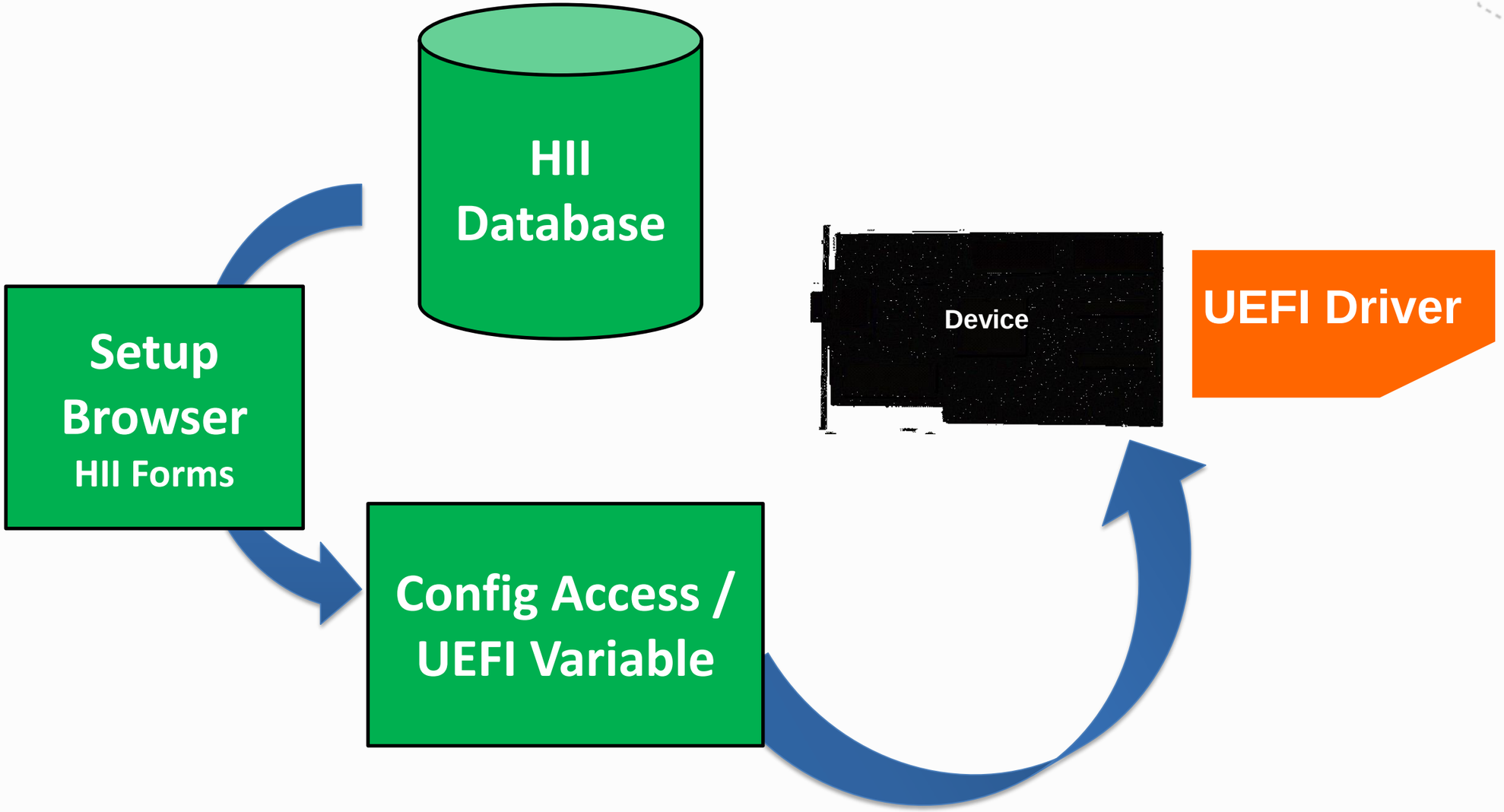
REST in UEFI

REST in UEFI

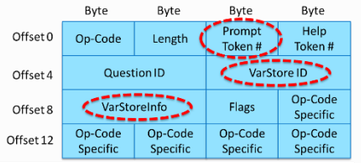
REST = REpresentational **S**tate **T**ransfer (architecture)
Most common client/server REST interface is HTTP



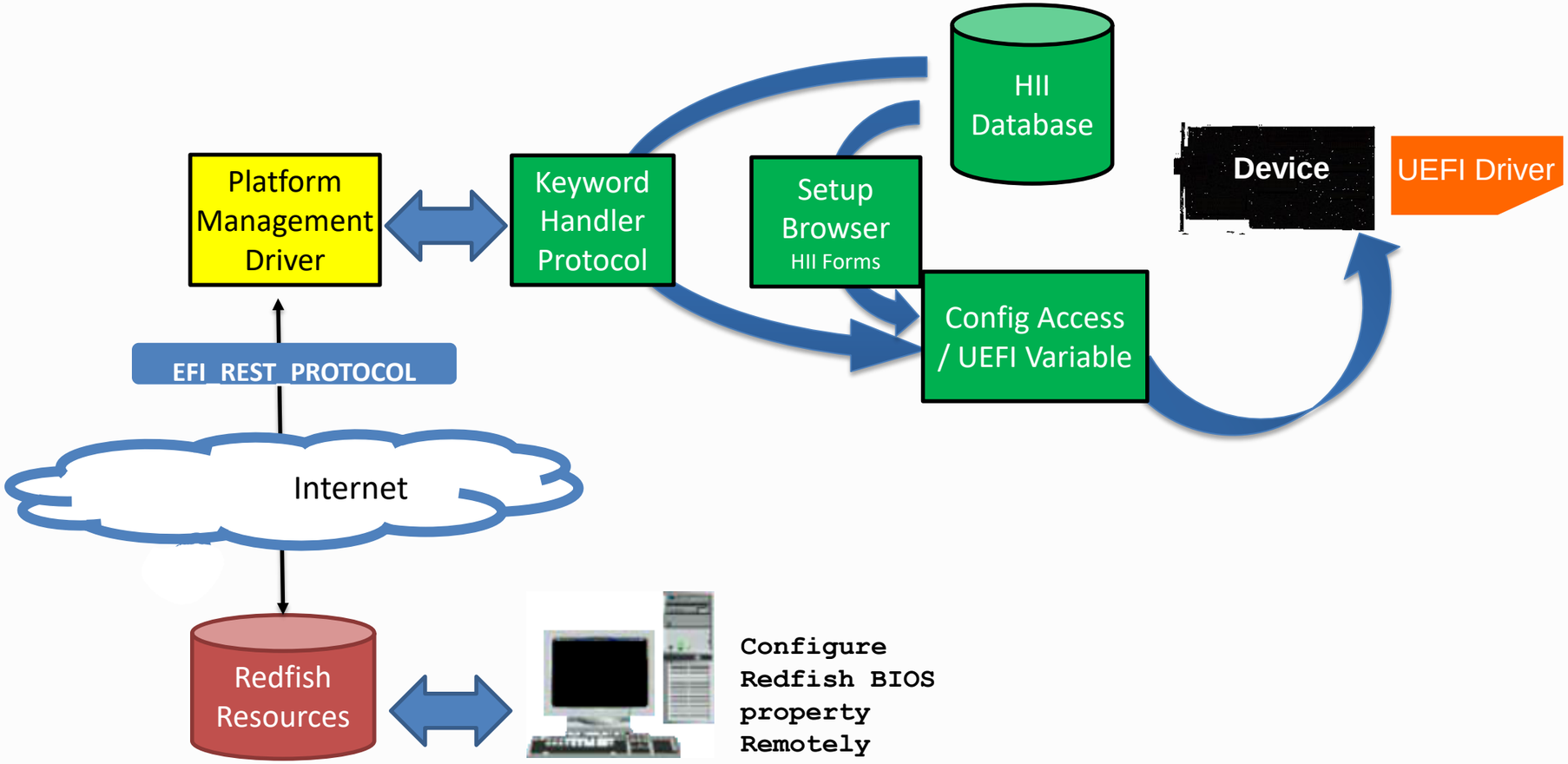
How does REST Relate to UEFI?



How does REST Relate to UEFI?

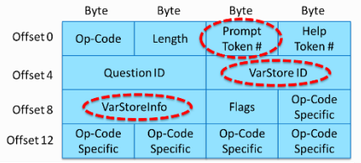



```
#x-UEFI Keyword "iSCSIInitiatorName"
```

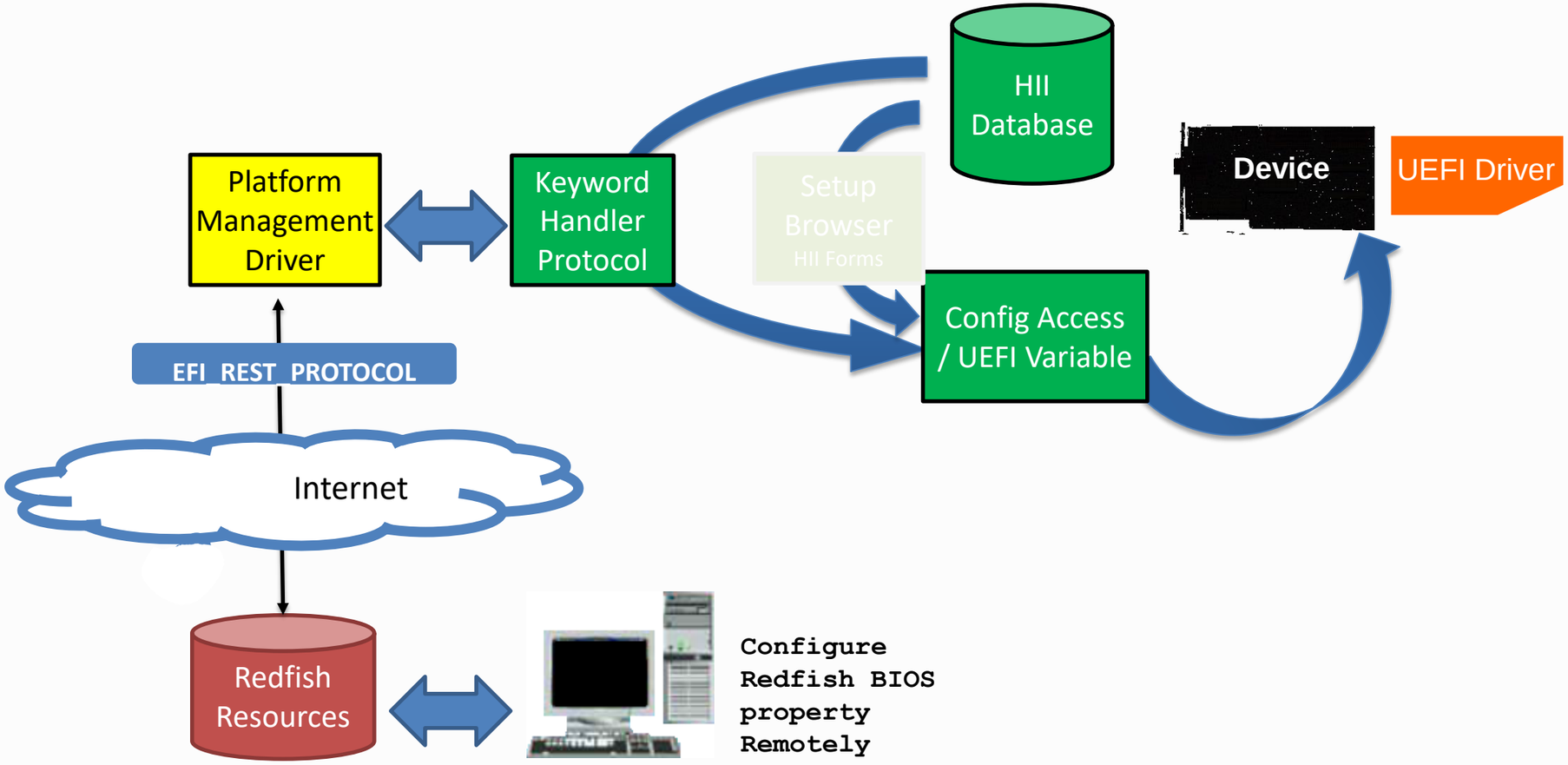



```
BIOS Redfish Attribute "iSCSIInitiatorName"
```

How does REST Relate to UEFI?




```
#x-UEFI Keyword "iSCSIInitiatorName"
```

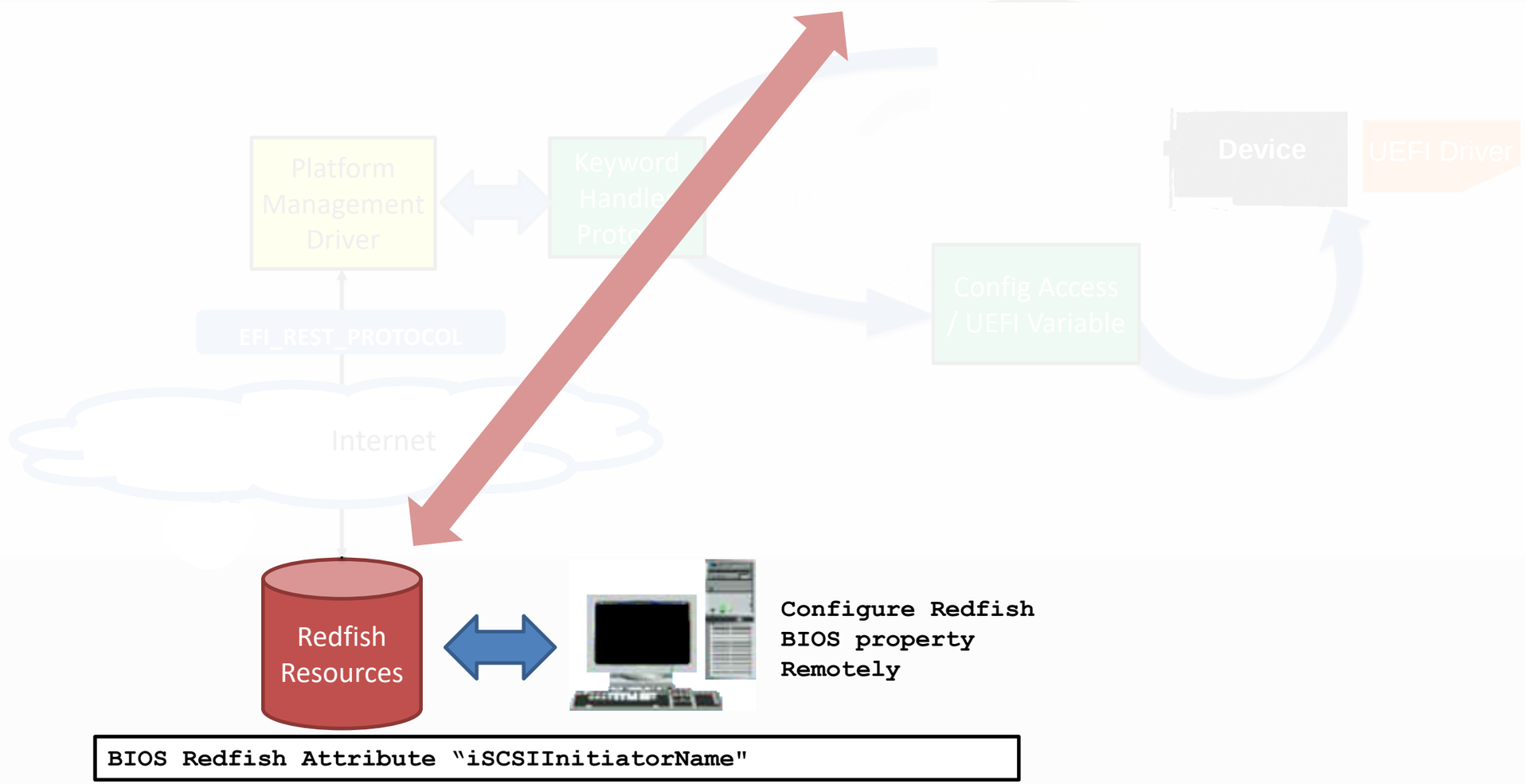



```
BIOS Redfish Attribute "iSCSIInitiatorName"
```

How does REST Relate to UEFI?

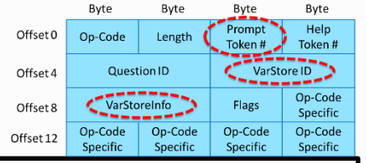
	Byte	Byte	Byte	Byte
Offset 0	Op-Code	Length	Prompt Token #	Help Token #
Offset 4	Question ID		VarStore ID	
Offset 8	VarStoreInfo		Flags	Op-Code Specific
Offset 12	Op-Code Specific	Op-Code Specific	Op-Code Specific	Op-Code Specific

```
#x-UEFI Keyword "iSCSIInitiatorName"
```

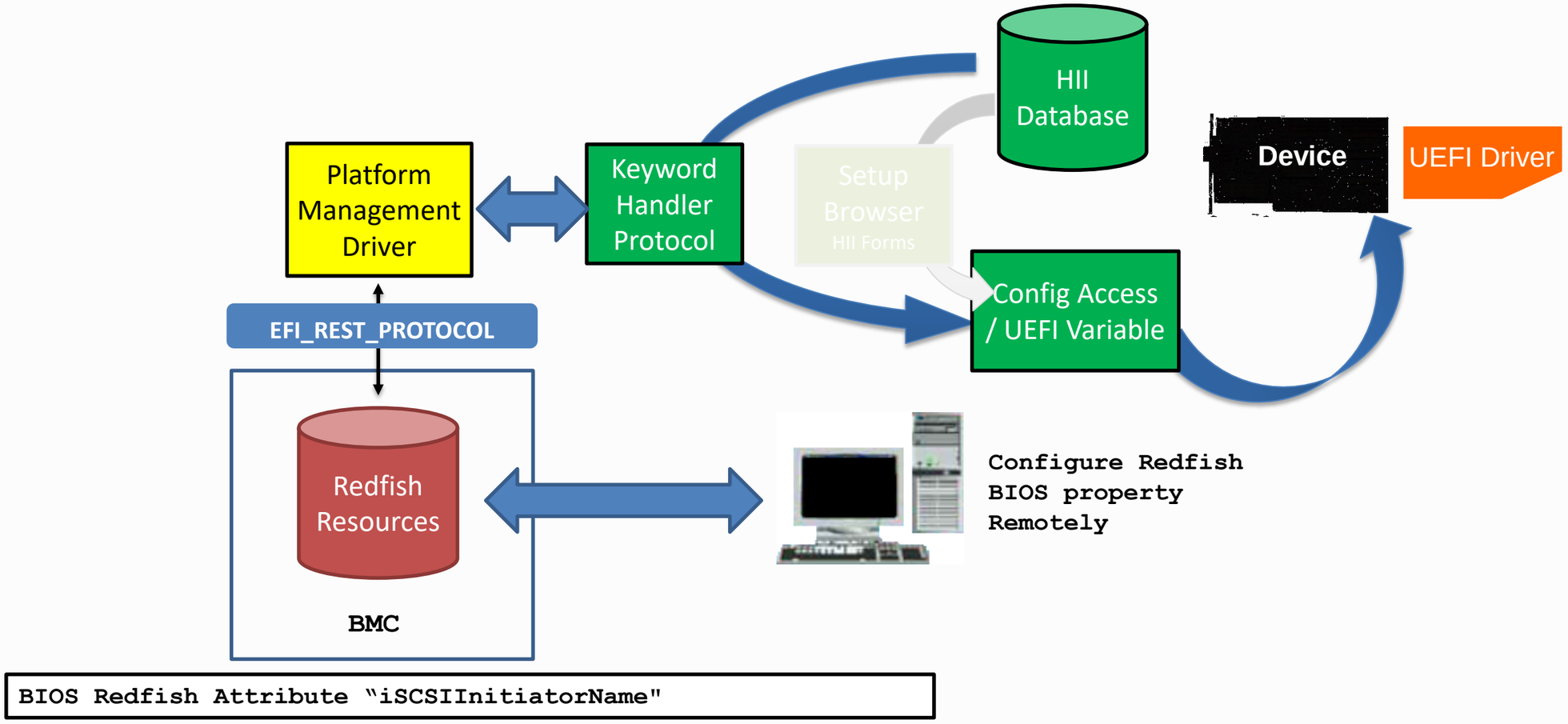



```
BIOS Redfish Attribute "iSCSIInitiatorName"
```

How does REST Relate to UEFI?




```
#x-UEFI Keyword "iSCSIInitiatorName"
```

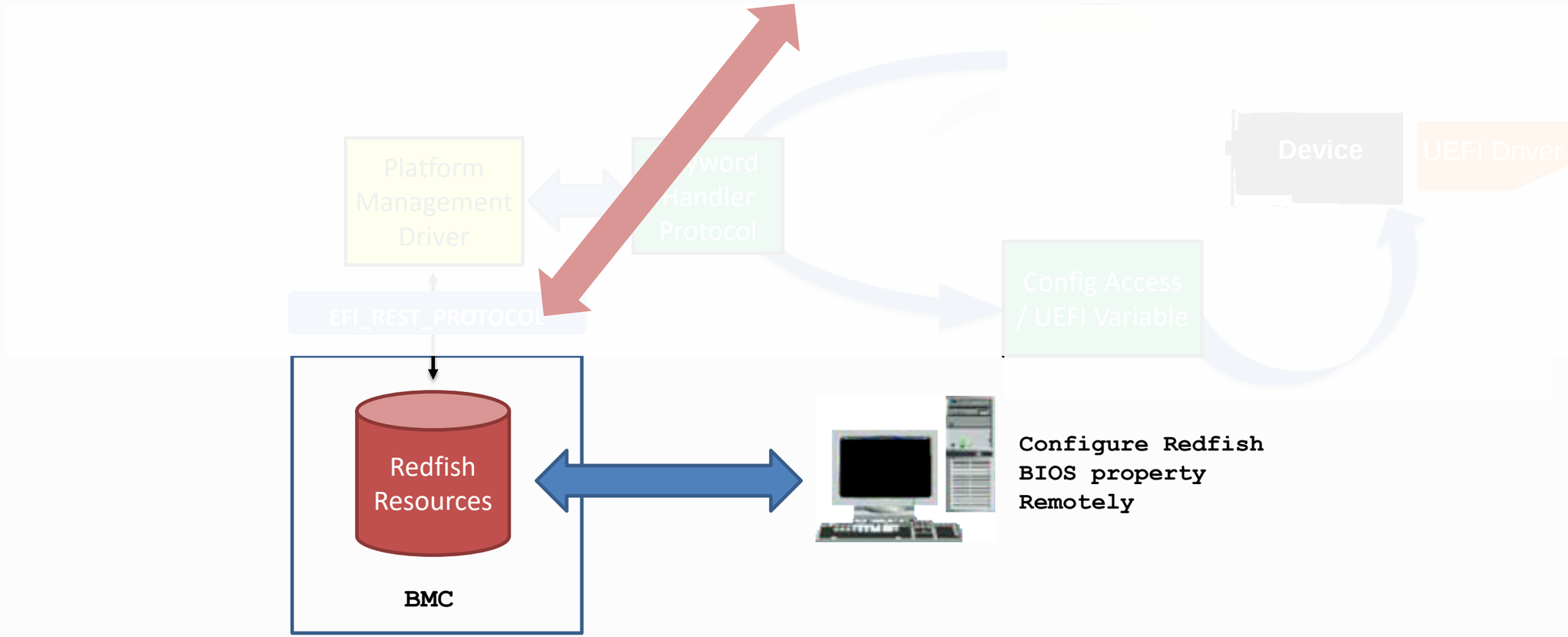



```
BIOS Redfish Attribute "iSCSIInitiatorName"
```

How does REST Relate to UEFI?

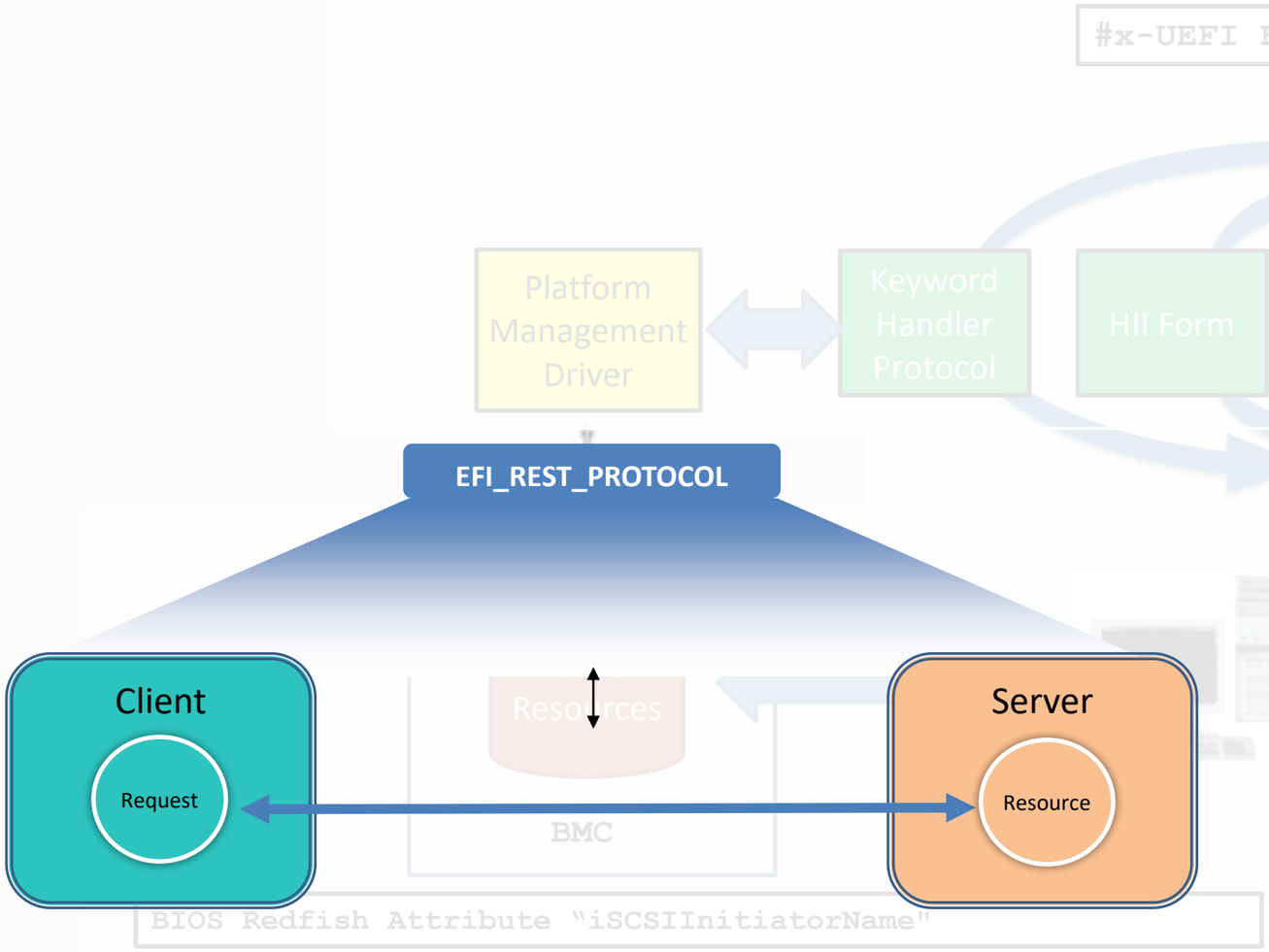
	Byte	Byte	Byte	Byte
Offset 0	Op-Code	Length	Prompt Token #	Help Token #
Offset 4	Question ID		VarStore ID	
Offset 8	VarStoreInfo		Flags	Op-Code Specific
Offset 12	Op-Code Specific	Op-Code Specific	Op-Code Specific	Op-Code Specific

```
#x-UEFI Keyword "iSCSIInitiatorName"
```




```
BIOS Redfish Attribute "iSCSIInitiatorName"
```

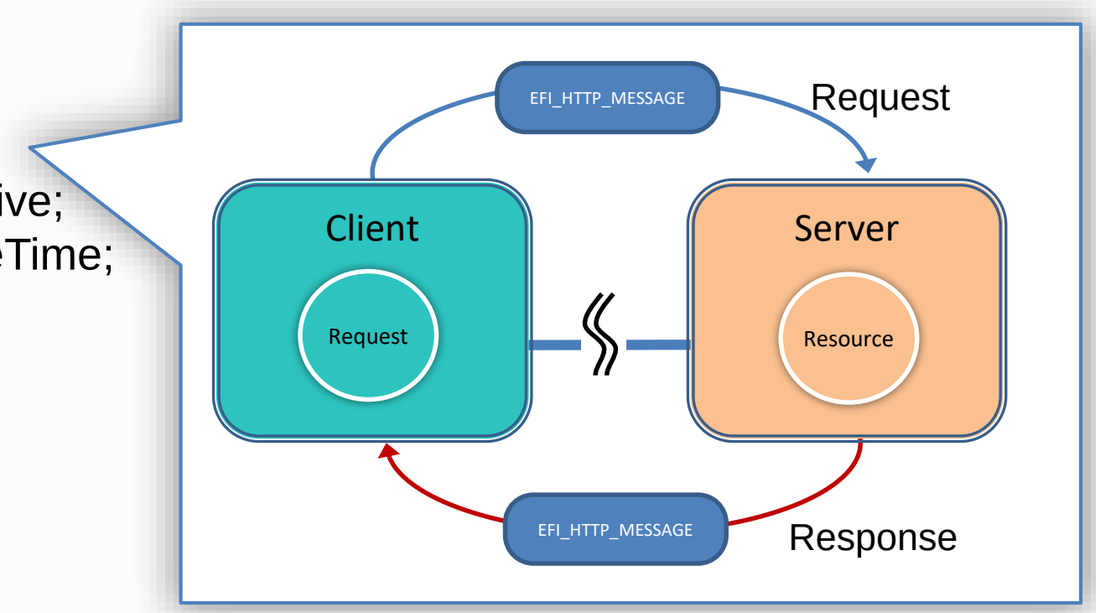
How does REST Relate to UEFI?



EFI REST Protocol in UEFI Spec v2.5

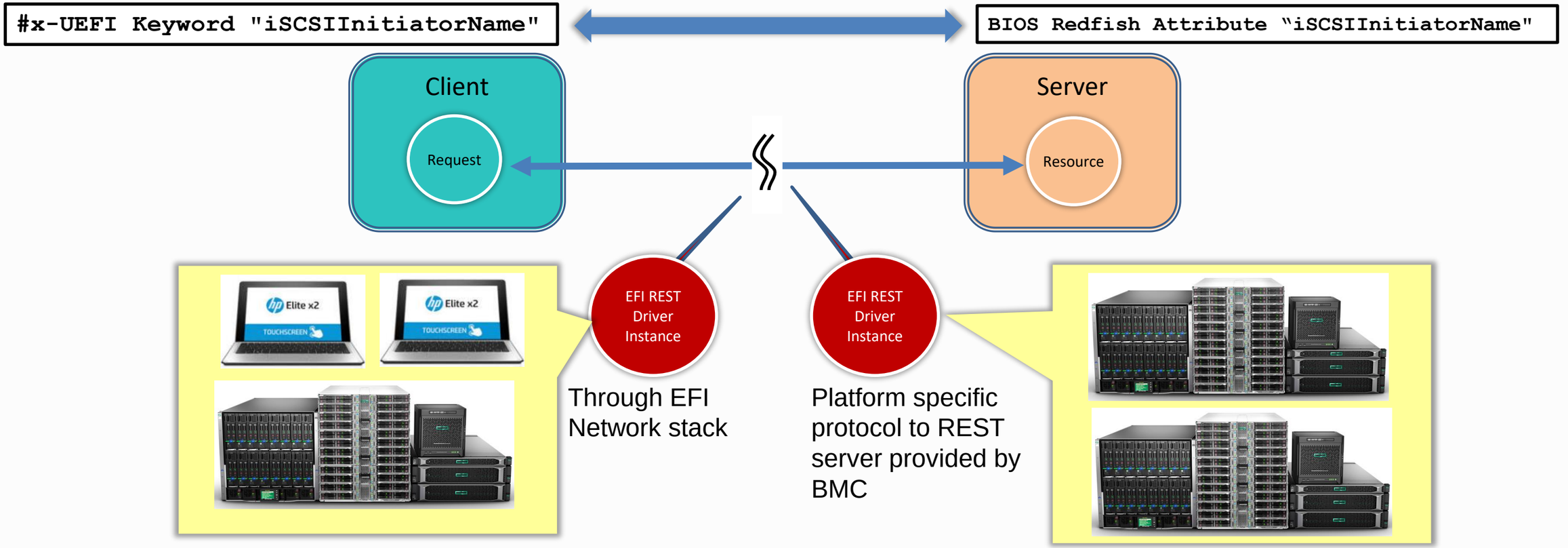
EFI REST Protocol use HTTP-like message as the format of REST service request and response

```
typedef struct _EFI_REST_PROTOCOL {
    EFI_REST_SEND_RECEIVE SendReceive;
    EFI_REST_GET_TIME         GetServiceTime;
} EFI_REST_PROTOCOL;
```



EFI REST Protocol in UEFI Spec v2.5

EFI REST driver instances may use different underlying protocol to communicate with REST server.



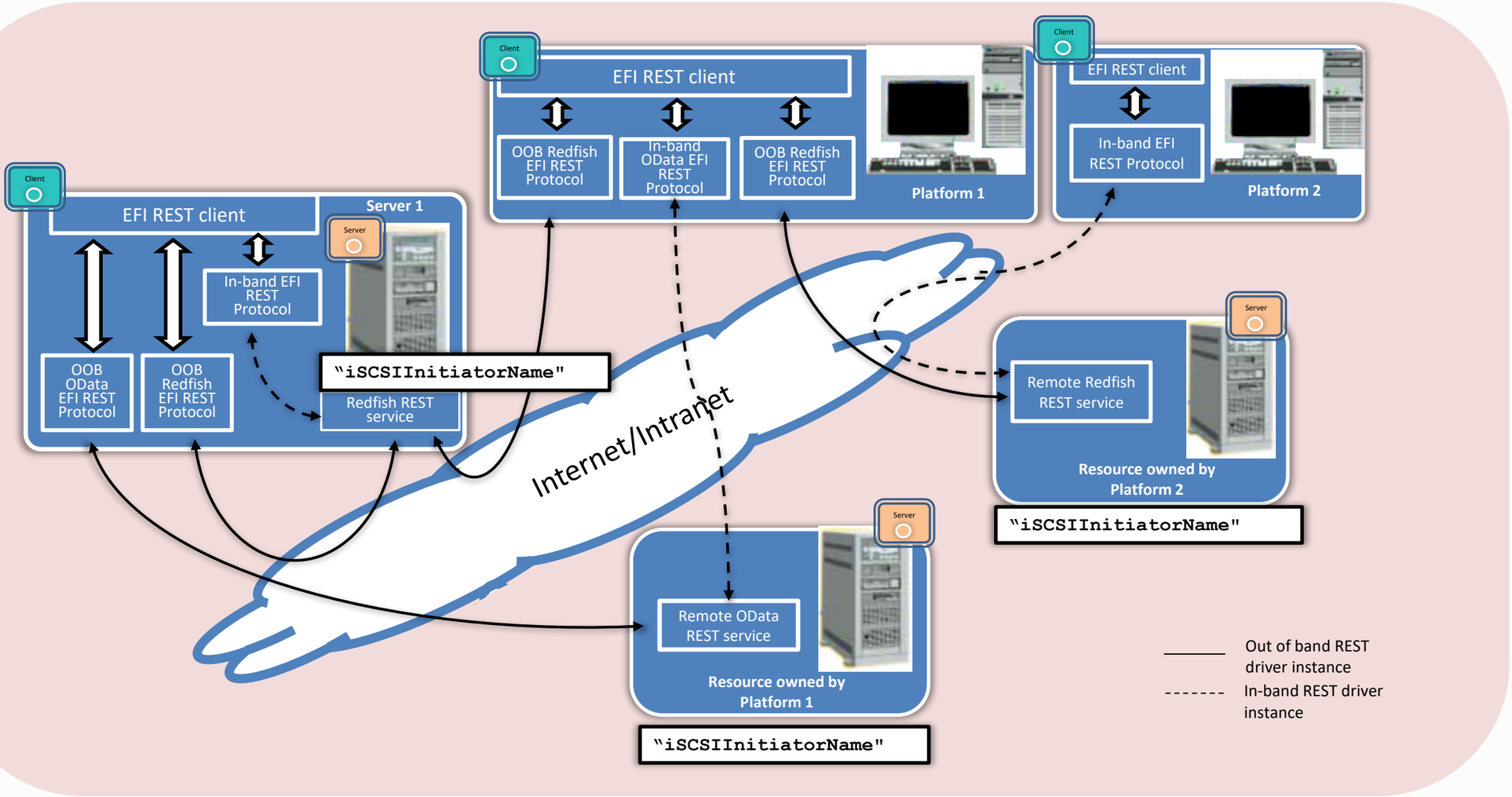
Insufficient Capabilities in ‘EFI REST’ and Proposals for Enhancement

Are ‘EFI REST’ Capabilities Sufficient?

- Are ‘EFI REST’ capabilities sufficient for UEFI Manageability?
- Are ‘EFI REST’ capabilities sufficient for the modern REST Services?

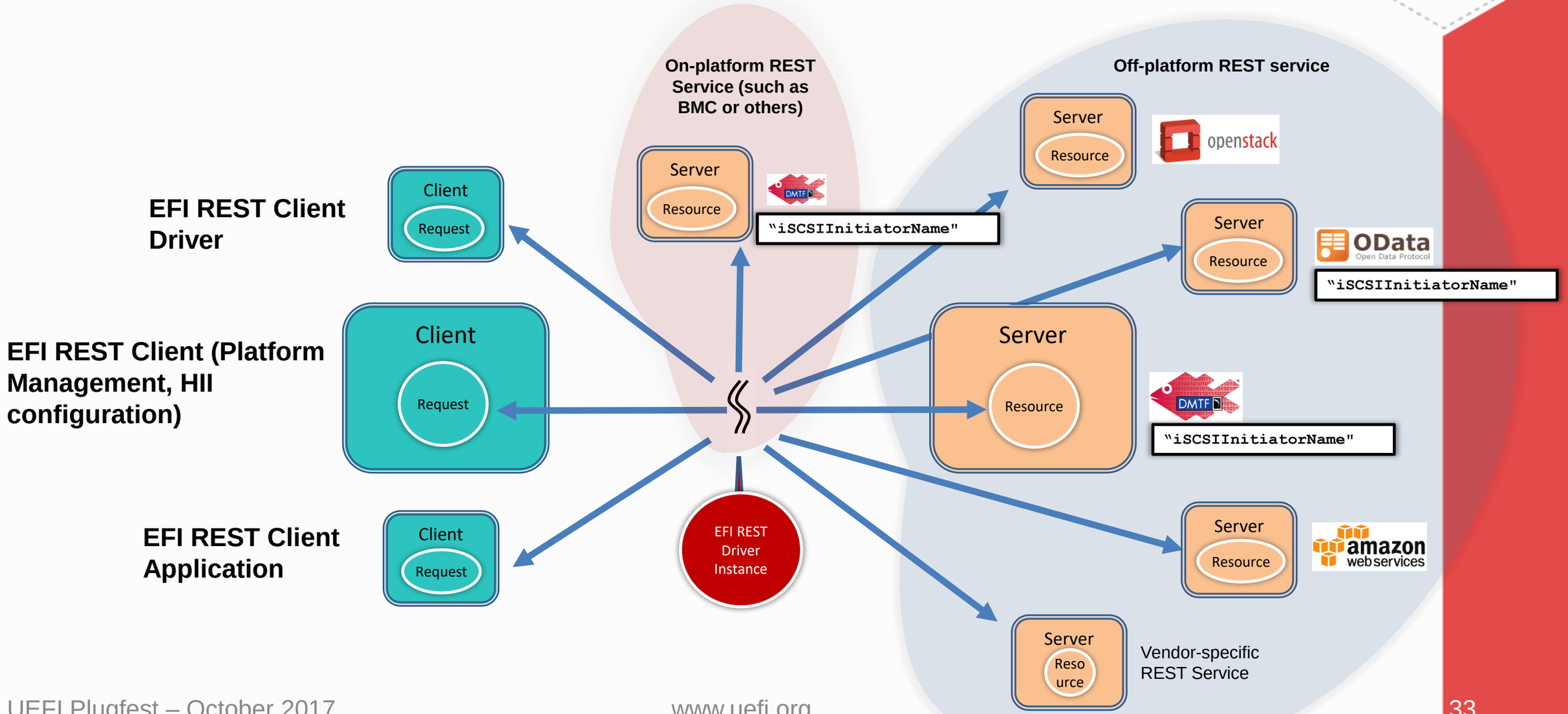
Are ‘EFI REST’ Capabilities Sufficient?

How can we support a variety of in-band/out of band UEFI management models as shown in this figure?



Are ‘EFI REST’ Capabilities Sufficient?

How can we support On-platform/Off-Platform REST services for EFI REST clients?



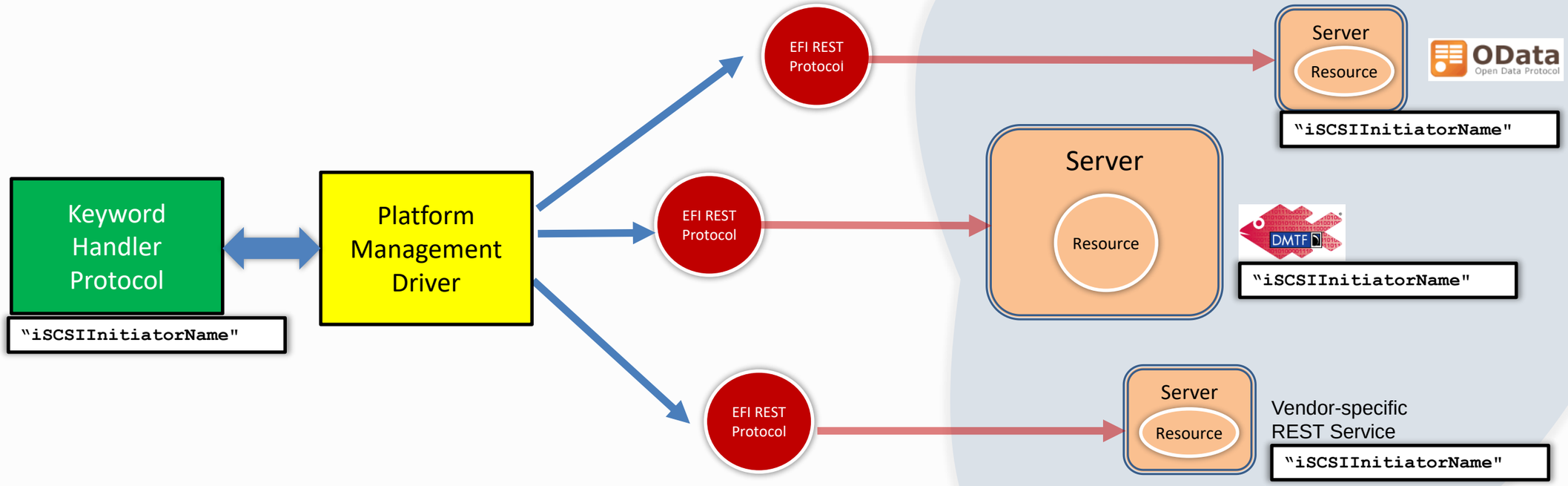
Suggestions for EFI REST Protocol Enhancements

- Multiple EFI REST driver instances for different REST services
- The information of REST service
- The location of REST service
- Asynchronous request and response
- Address the events from REST service

Suggestions for EFI REST Protocol Enhancements

Multiple EFI REST driver instances for different REST services

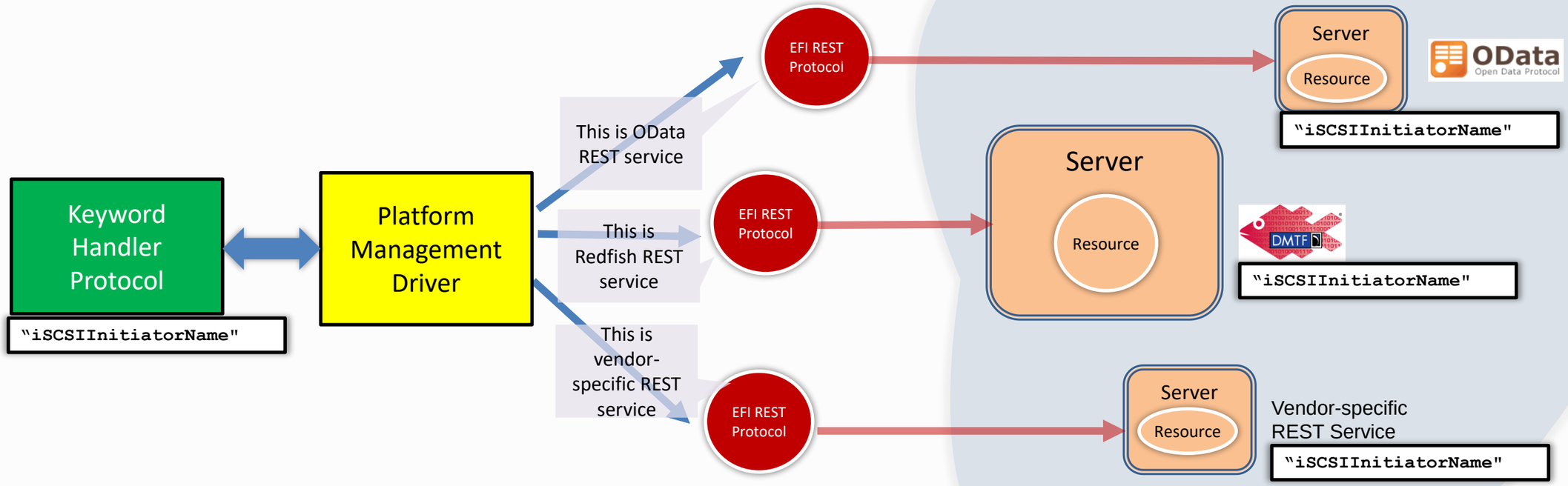
Multiple EFI REST driver instances can be installed on system for communicating to different REST services. The EFI REST client drivers and applications can use the specific EFI REST protocol to access to REST service.



Suggestions for EFI REST Protocol Enhancements

The information of REST service?

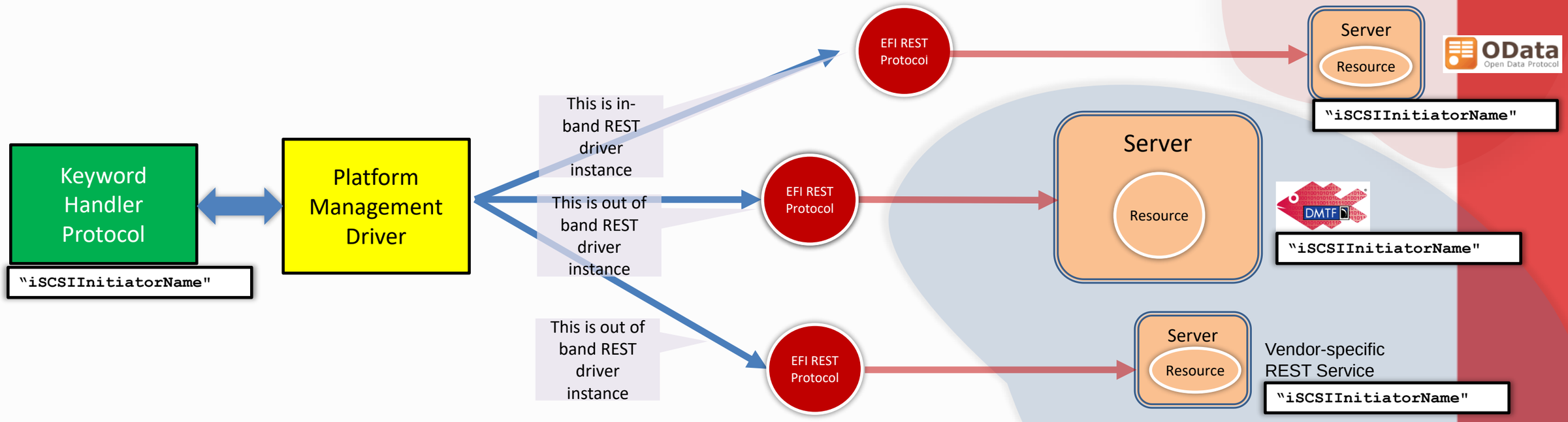
Each EFI REST driver instance has to provide the information about REST service it supports.



Suggestions for EFI REST Protocol Enhancements

The location of REST service?

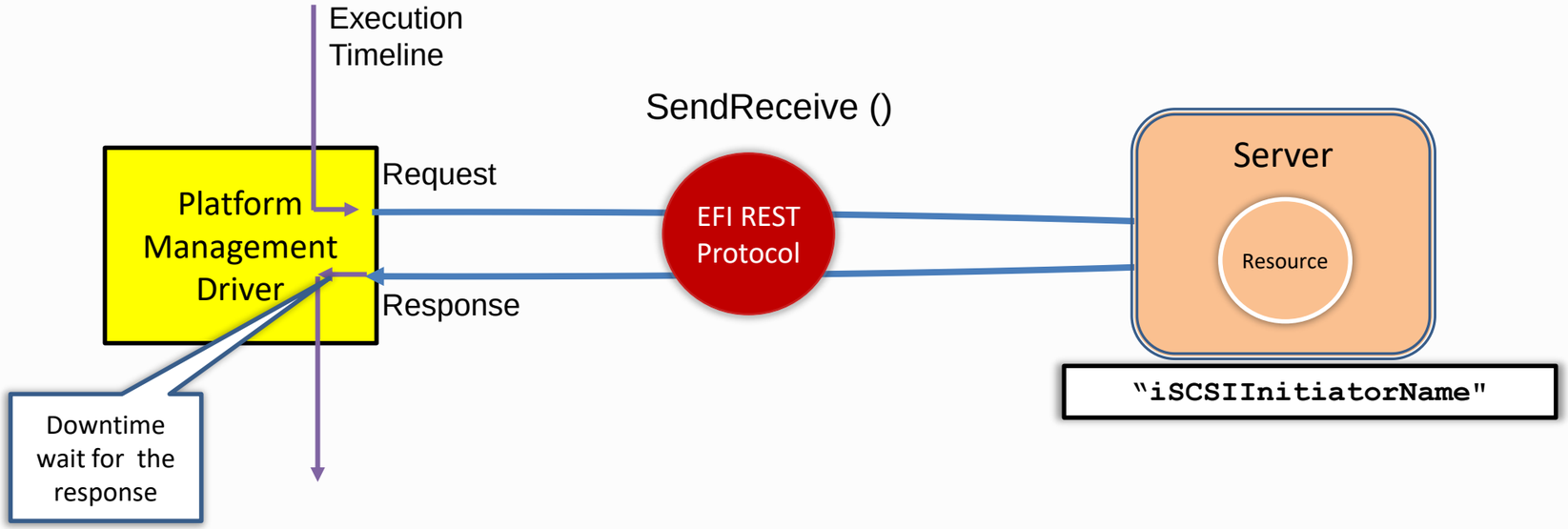
Each EFI REST driver instance has to provide the information about the location of REST service.



Suggestions for EFI REST Protocol Enhancements

Synchronous request and response leads to a downtime

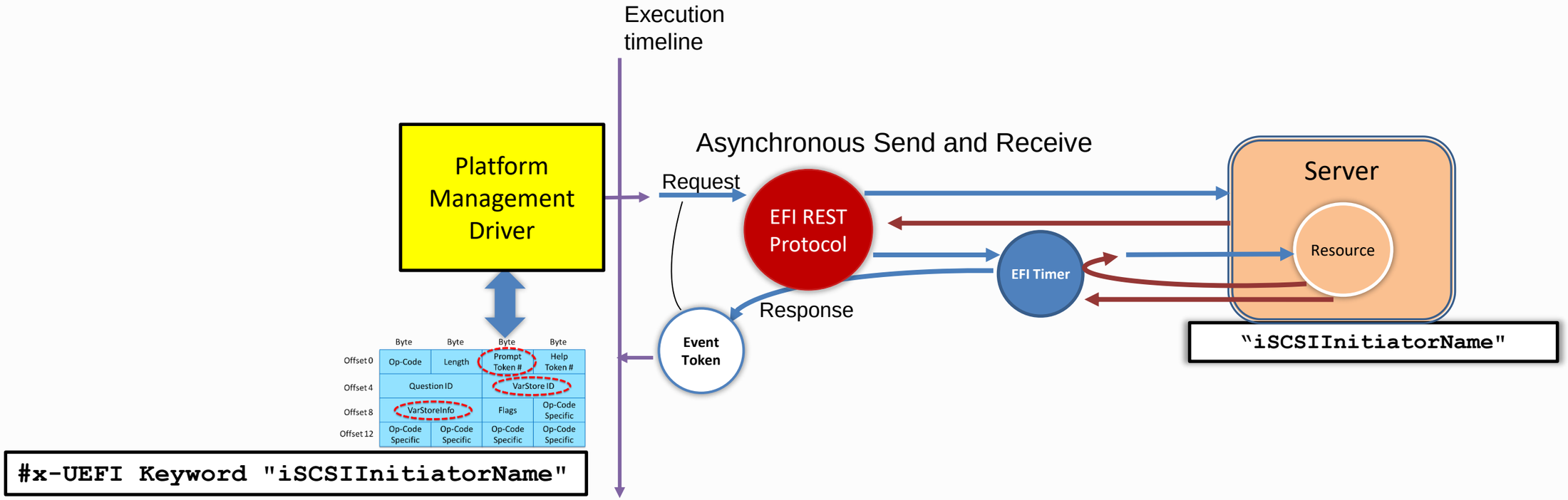
SendReceive() function provided in EFI REST protocol is synchronous transfer.
EFI REST client has to wait for the response once it sends the request out.



Suggestions for EFI REST Protocol Enhancements

Asynchronous request and response

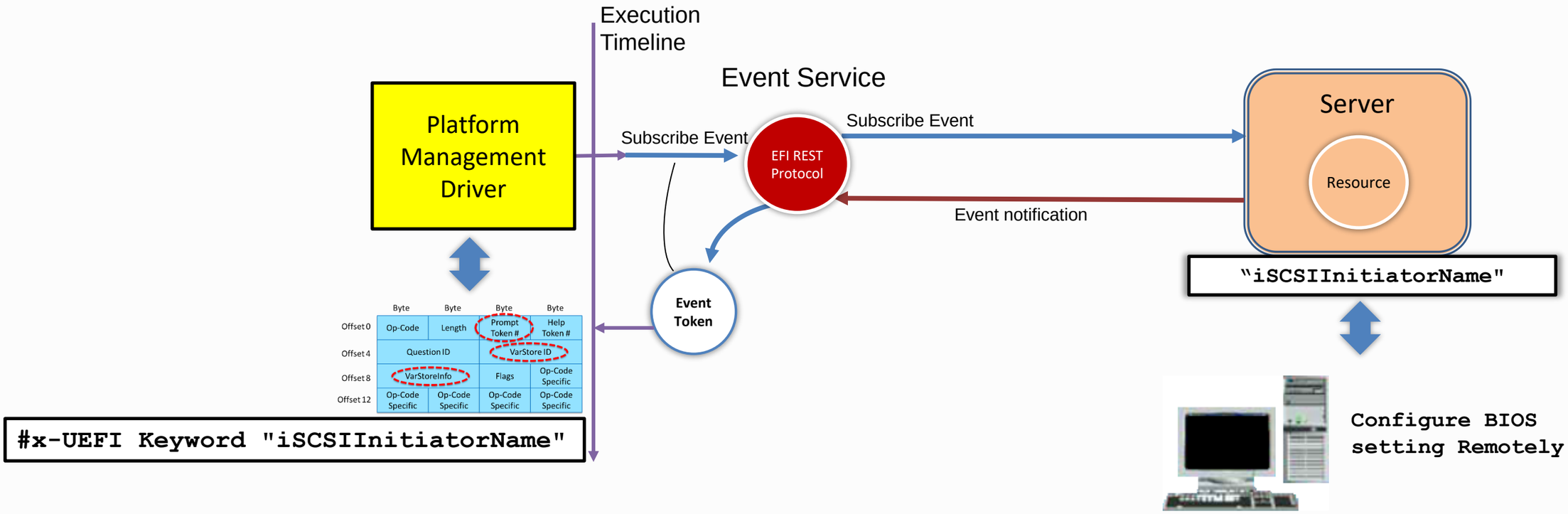
Support asynchronous SendReceive() function to provide efficient REST request and response. Also provides the better user experience when users use EFI REST client application



Suggestions for EFI REST Protocol Enhancements

Address the events from REST service

Most of REST service support event capability to indicate the changes of resource. User can register the event for certain type resource change such as resource created, modified, deleted and etc.



Summary & Call to Action

Summary

- UEFI provides interfaces for improved manageability
  - (HII, x-UEFI, Keyword handle protocol, EFI REST protocol)
- Improvements to REST are required for adoption of standardized manageability

Call to Action

- Define and register more x-UEFI keywords
- Enhance current EFI REST Protocol for flexible, scalable and user friendly interfaces for EFI REST Clients
  - Asynchronous request/response in the enhanced EFI REST Protocol
  - Event subscription for monitoring resource changes in enhanced EFI REST Protocol
  - Provide sufficient information for target REST service

Thanks for attending the Fall 2017
UEFI Plugfest

For more information on the UEFI Forum and UEFI Specifications, visit
http://www.uefi.org

presented by

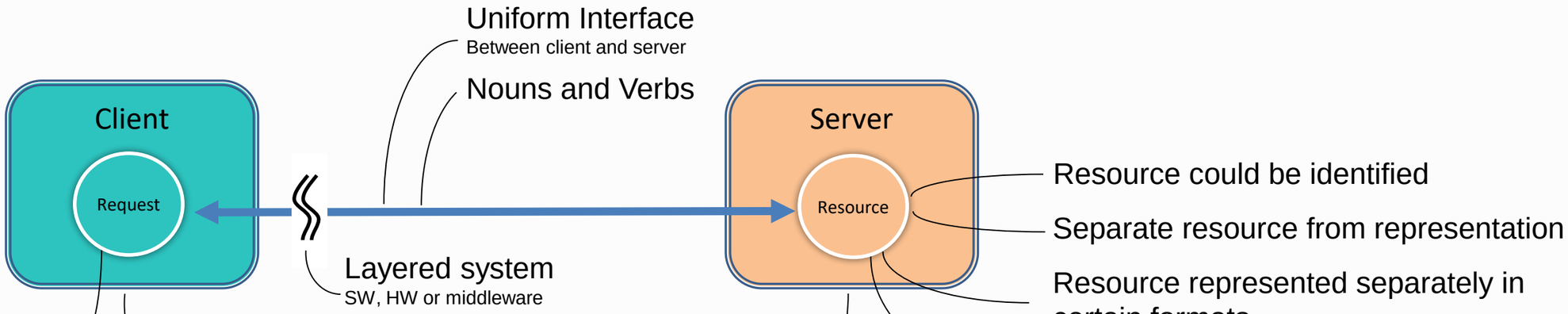
Hewlett Packard Enterprise

intel

Backup

What is REST?

REpresentational **S**tate **T**ransfer, which is a software architecture style. There are some constraints applied to this architecture.



No client context stored in server. The necessary state is contained within request itself.

*The request contains the unique identifier of resource and the request body contains the state (or state change) of that resource

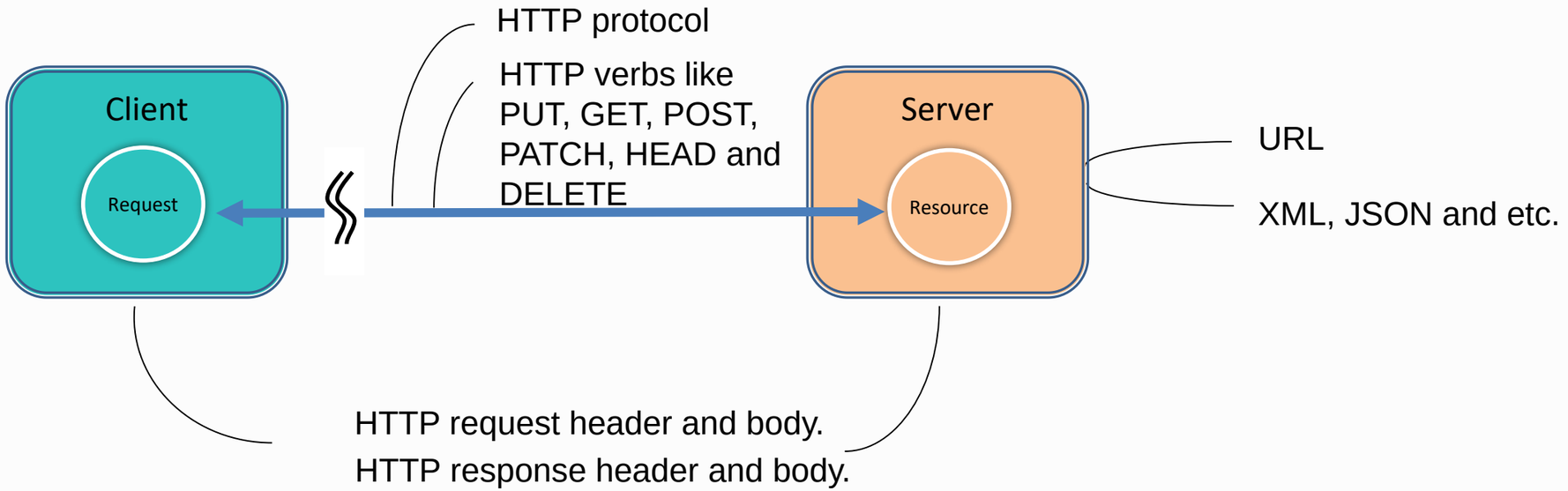
*The state after server processed the request is retuned in response body.

Client and Server, separating the user interface from resource. Which improves portability and scalability.

* Client don't have to concern the data storage
* Server don't have to concern the user interface and user state.

REST in HTTP

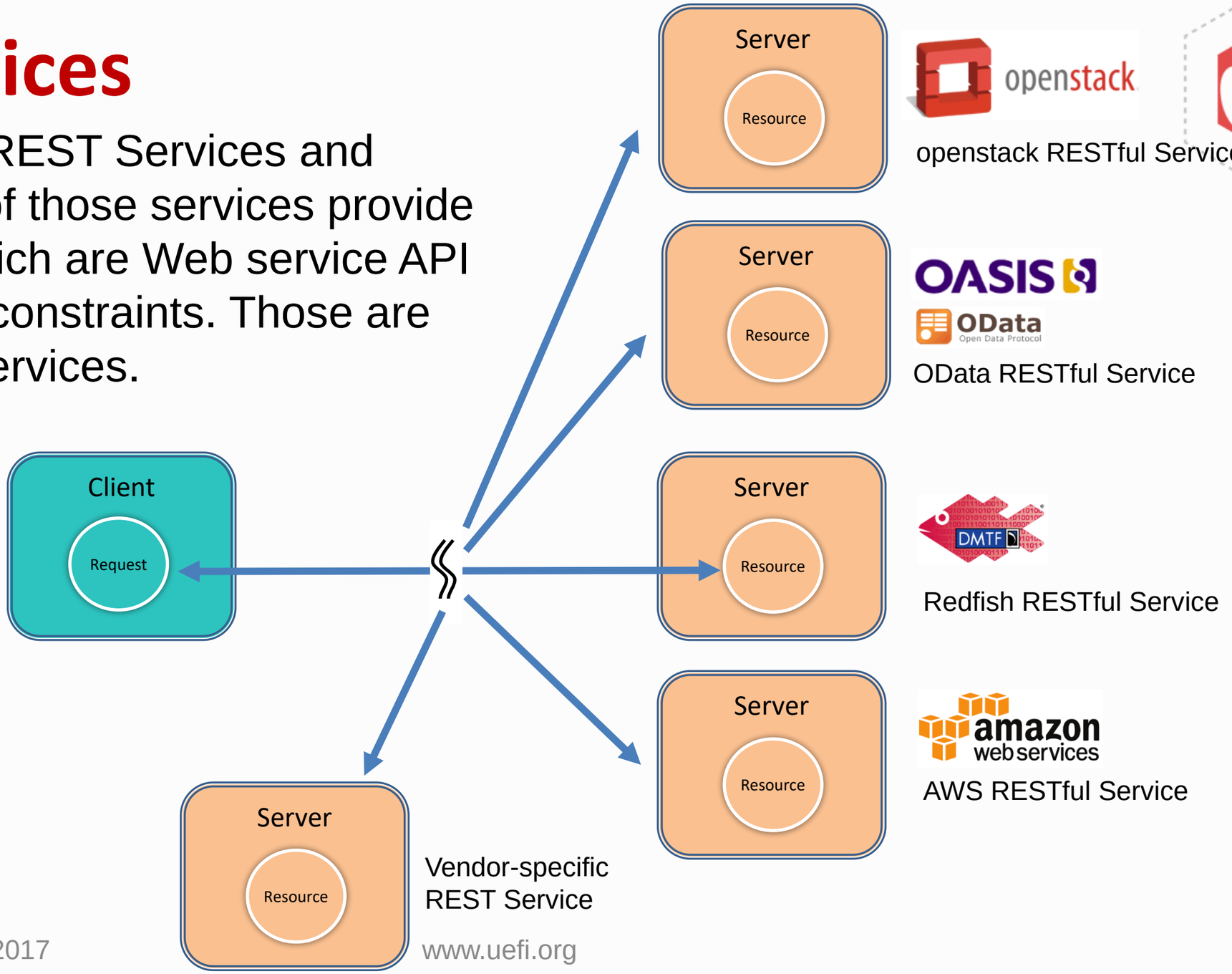
HTTP is not the only interface (protocol) for REST architecture. However, it's most commonly used in Web REST service. Any transport interface which is unified to REST service and have the well defined verb to manipulate resource could be the interface between REST server and REST client.



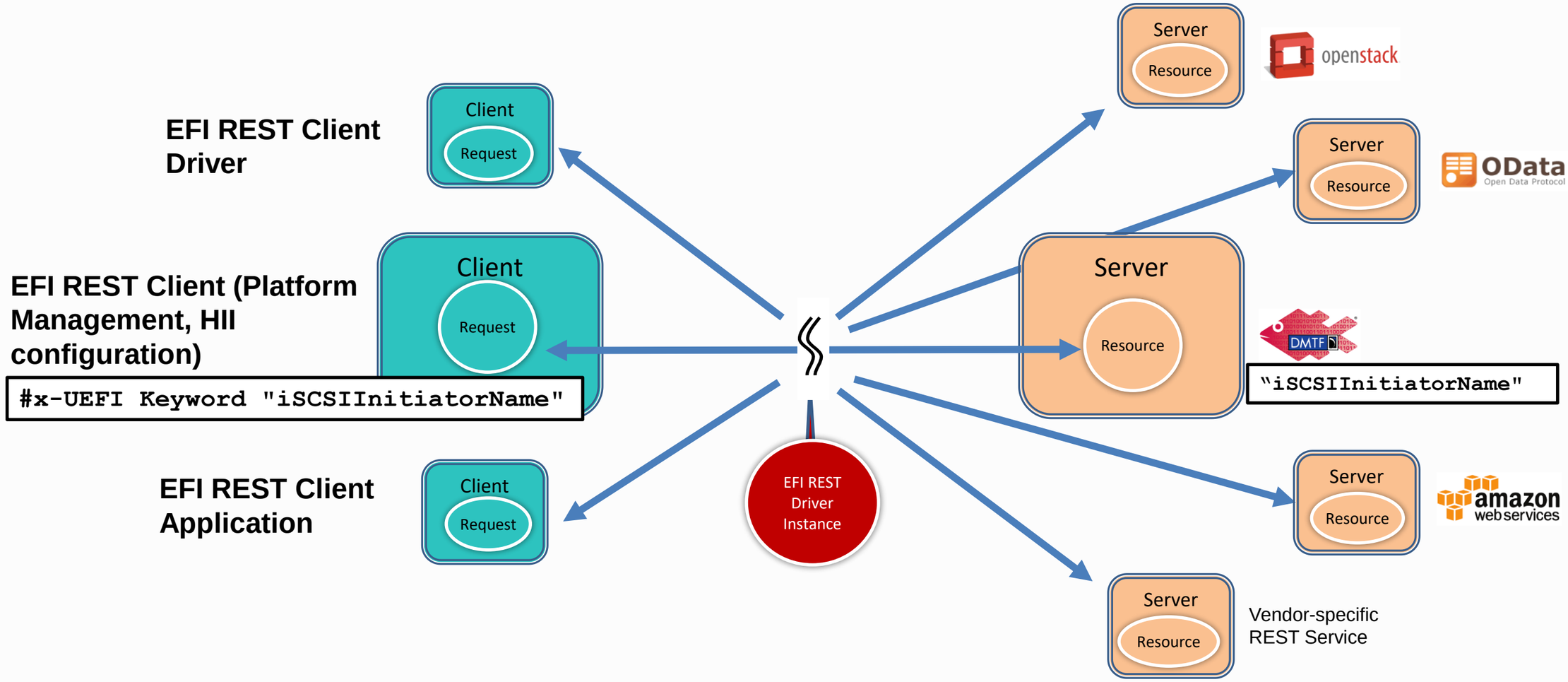
REST Client and REST Services

REST Services

There are many REST Services and Providers. Most of those services provide RESTful APIs which are Web service API adhere to REST constraints. Those are called RESTful services.



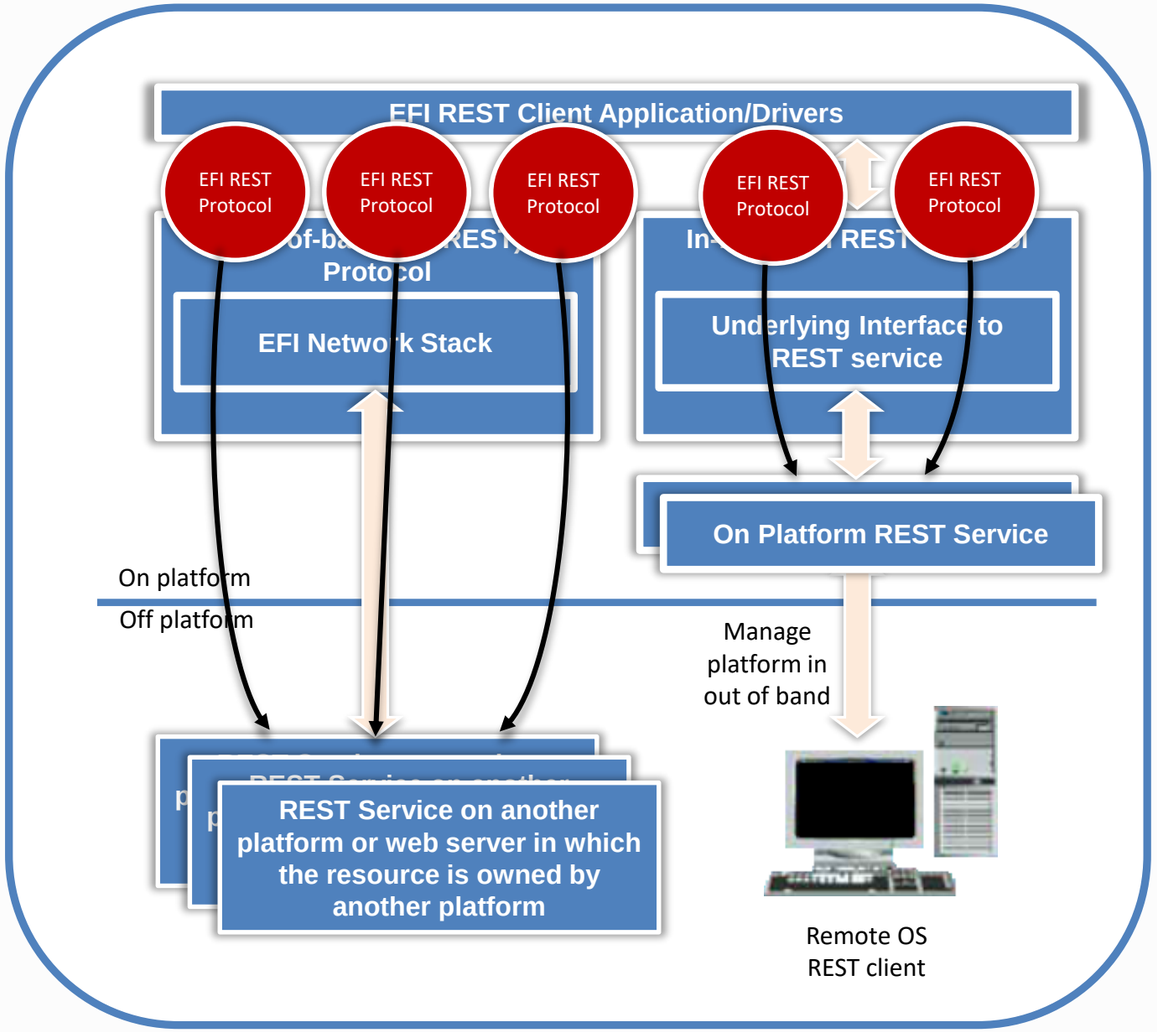
REST Service Server and EFI REST Client



Suggestions for EFI REST Protocol Enhancements

The location of REST service provided by specific EFI REST driver instance

Each EFI REST driver instance has to provide the information about the location of REST service it supports.



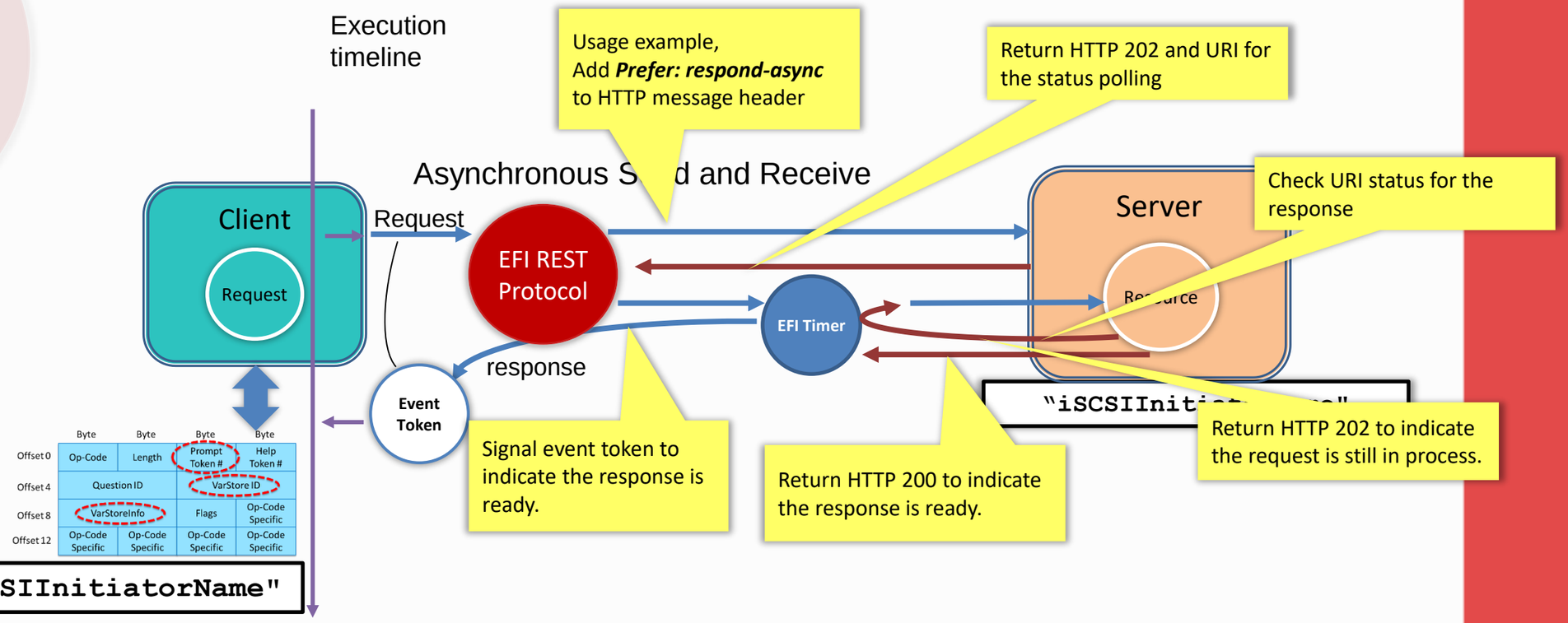
Suggestions for EFI REST Protocol Enhancements

Asynchronous request and response

Enhancement of EFI REST Protocol

SendReceive() function provided in EFI REST protocol is synchronous transfer. EFI REST client has to wait for the response once it sends the request out.

Support asynchronous SendReceive() function to provide efficient REST request and response. Also provide the better user experience when users use EFI REST client application

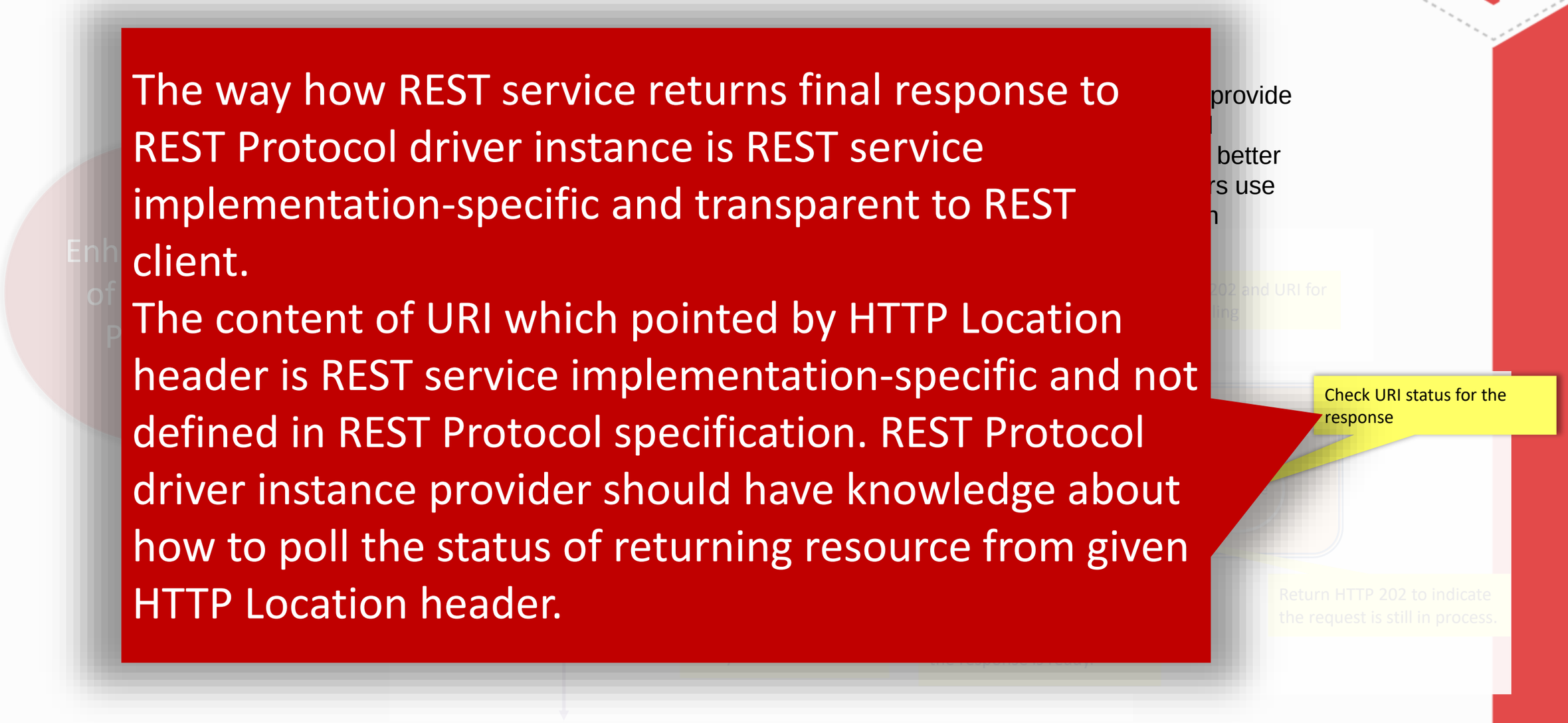


Suggestions for EFI REST Protocol Enhancements

Asynchronous request and response

The way how REST service returns final response to REST Protocol driver instance is REST service implementation-specific and transparent to REST client.

The content of URI which pointed by HTTP Location header is REST service implementation-specific and not defined in REST Protocol specification. REST Protocol driver instance provider should have knowledge about how to poll the status of returning resource from given HTTP Location header.

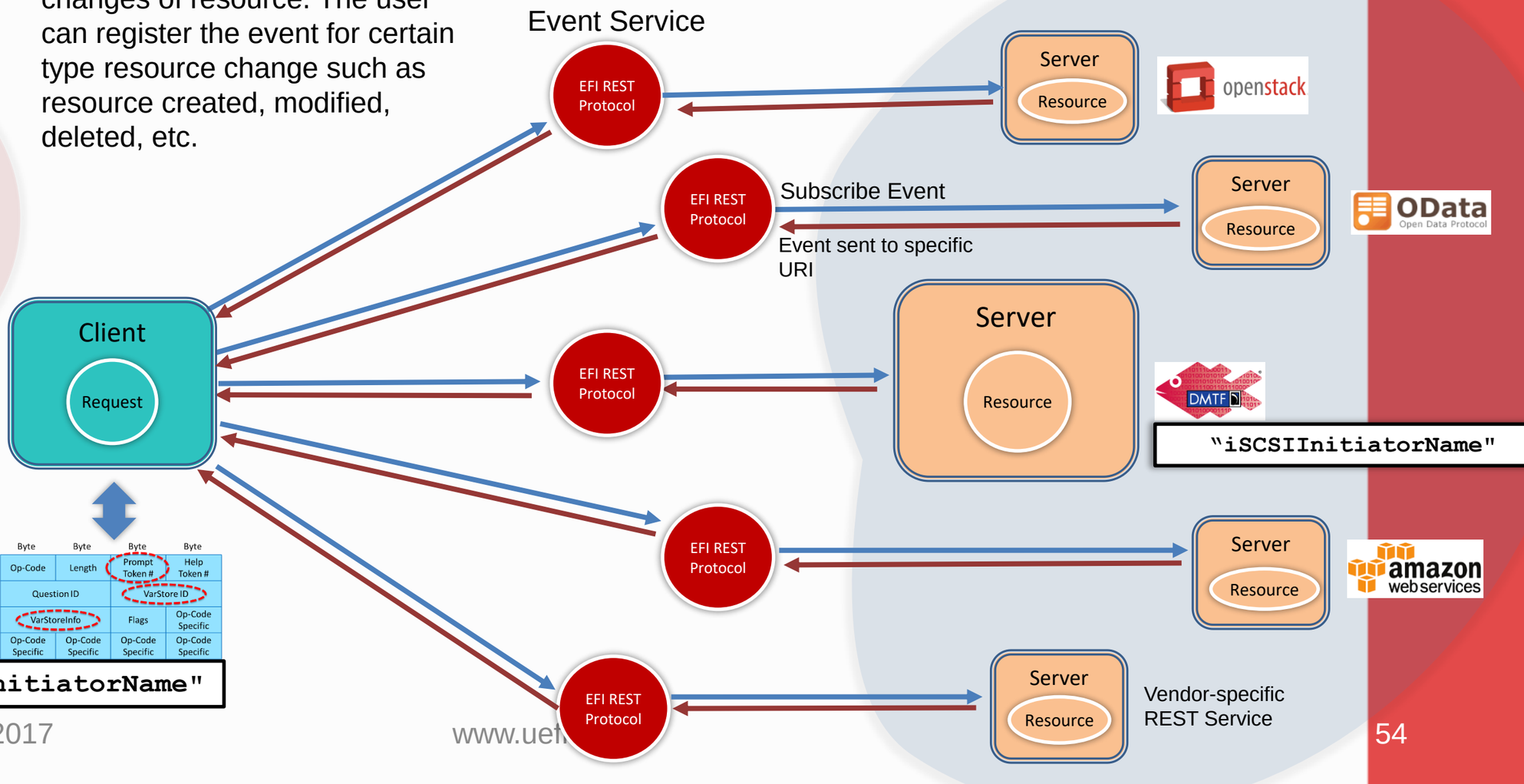


Suggestions for EFI REST Protocol Enhancements

Address the events from REST service

Most of REST service support event capability to indicate the changes of resource. The user can register the event for certain type resource change such as resource created, modified, deleted, etc.

Enhancement of EFI REST Protocol



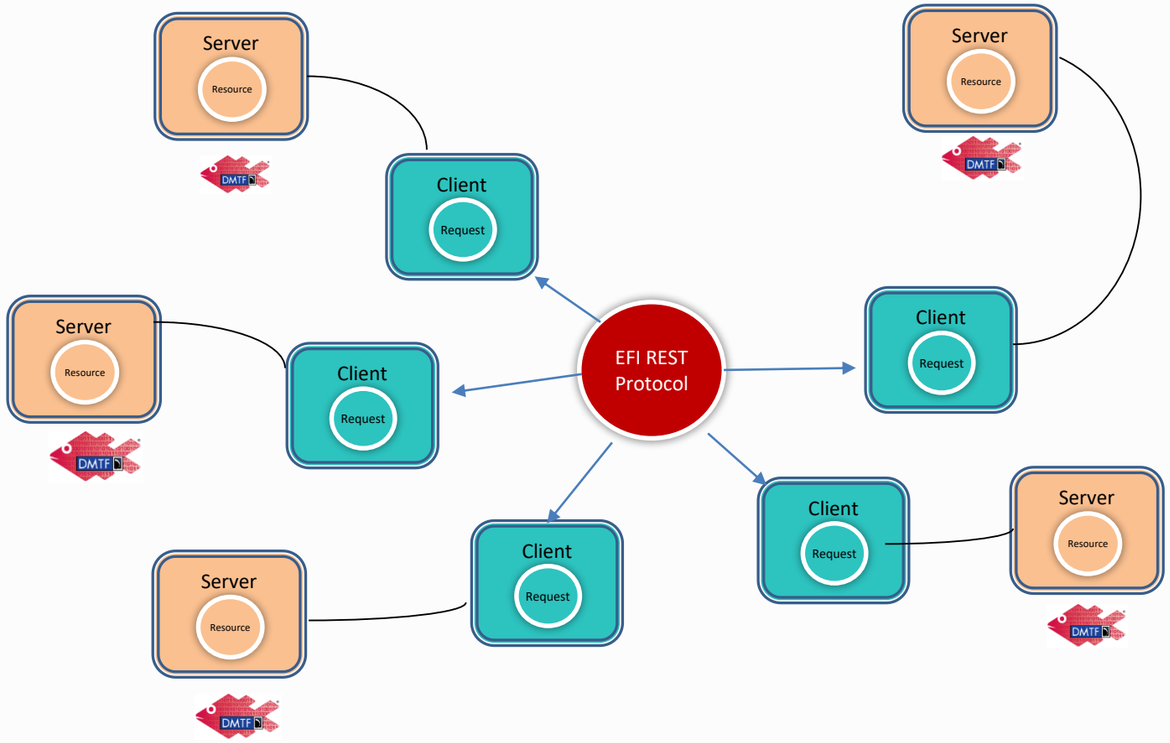

```
#x-UEFI Keyword "iSCSIInitiatorName"
```

Suggestions for EFI REST Protocol Enhancements

Multiple EFI REST clients to access to different REST service

Each EFI REST child instance communicates to different REST services.

Enhancement of EFI REST Protocol



Suggestions for EFI REST Protocol Enhancements

Multiple EFI REST clients to access to one REST service

Enhancement of EFI REST Protocol

Each EFI REST child instance communicates to different REST services.

Or multiple EFI REST child instances access to the same REST service.