

# Role of scripting API & management interface

May 18<sup>th</sup>, 2017

Kazuaki Nimura  
Fujitsu Laboratories

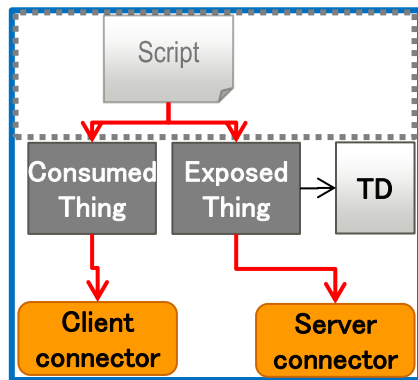
## Differences

### Scripting API:

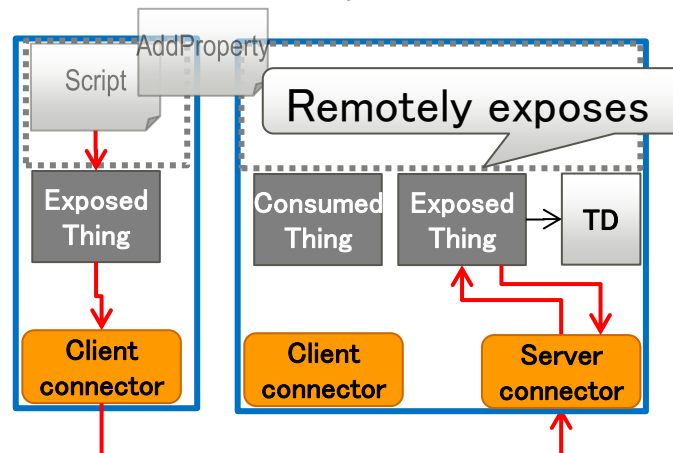
- Local execution of Scripting API script.
- Remote execution of Scripting API script.

### Management Interface: Remote execution of management command or script.

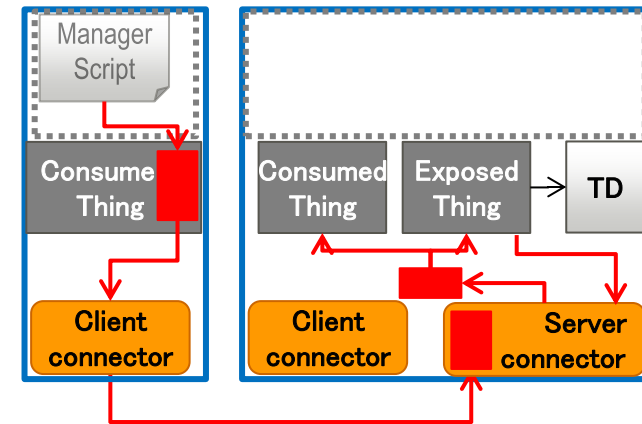
ExposeThing is limited to LocalThing ?  
ExposeThing can be controlled remotely?



(1) ScriptingAPI(Local)



(2) ScriptingAPI(Remote)

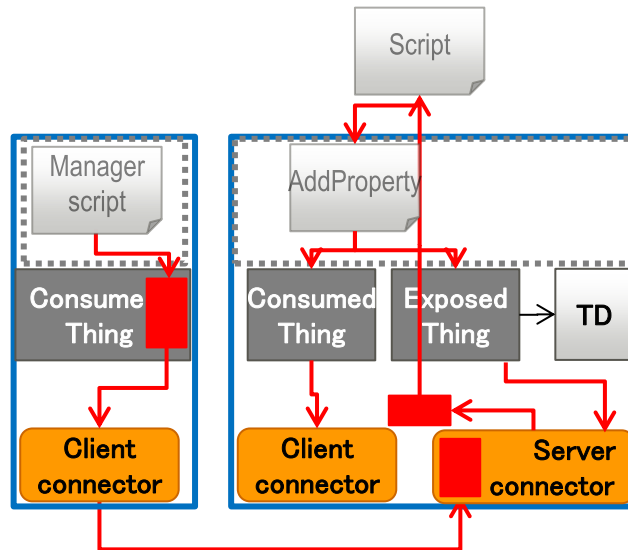


(3) Management interface

## ■ Management Interface:

- Can setup runtime by downloading a script and executed the script included scripting API e.g. AddProperty.
- (2) and (4) provides almost same outcome.

What is our target at all?

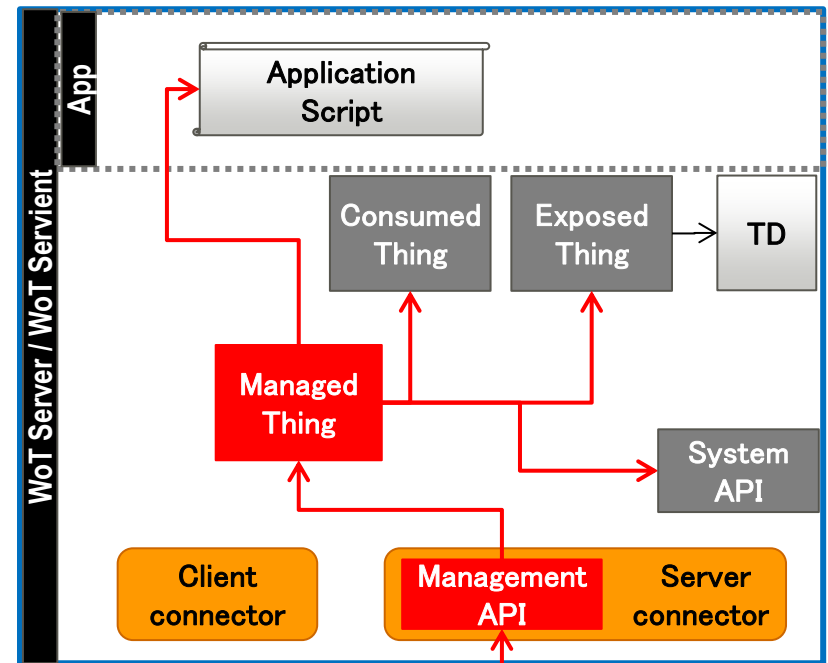


(4) Management interface (script download and execute)

# Management interface

- Management interface is underlying technology of Runtime.
- Management API and ManagedThing exists in a runtime (WoT Server or WoT Servient) and deals provisioning of runtime.
- ManagedThing works based on management commands.
- ManagedThing co-work with ScriptingAPI.
  - TD of ManagedThing provides information about what kind of management capabilities are available in the runtime.
- ManagedThing deals system API call.

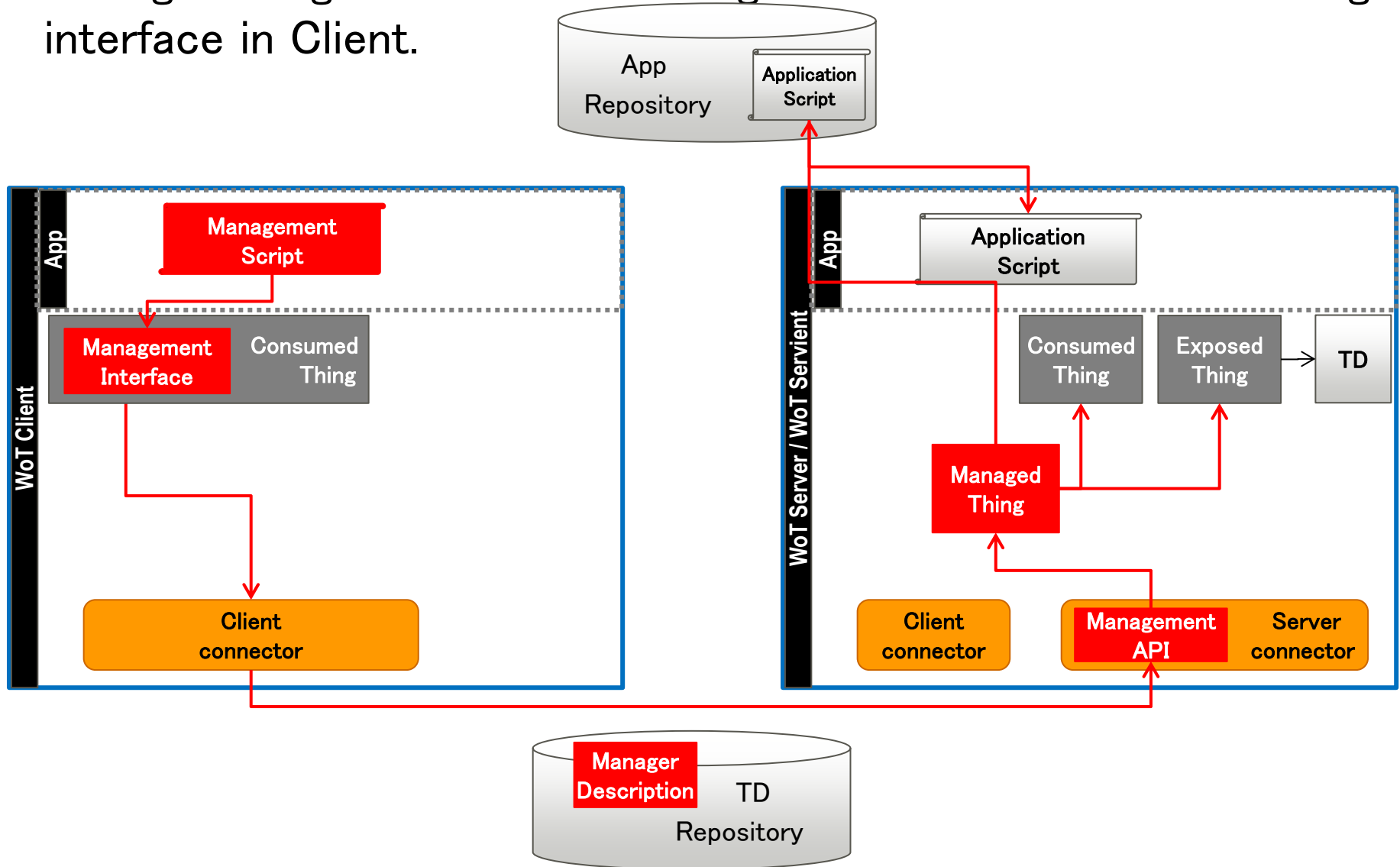
Management commands  
Management interface



Note: "Management script", "Management API", "Management Interface", "ManagedThing," and "Manager Description" are tentative names.

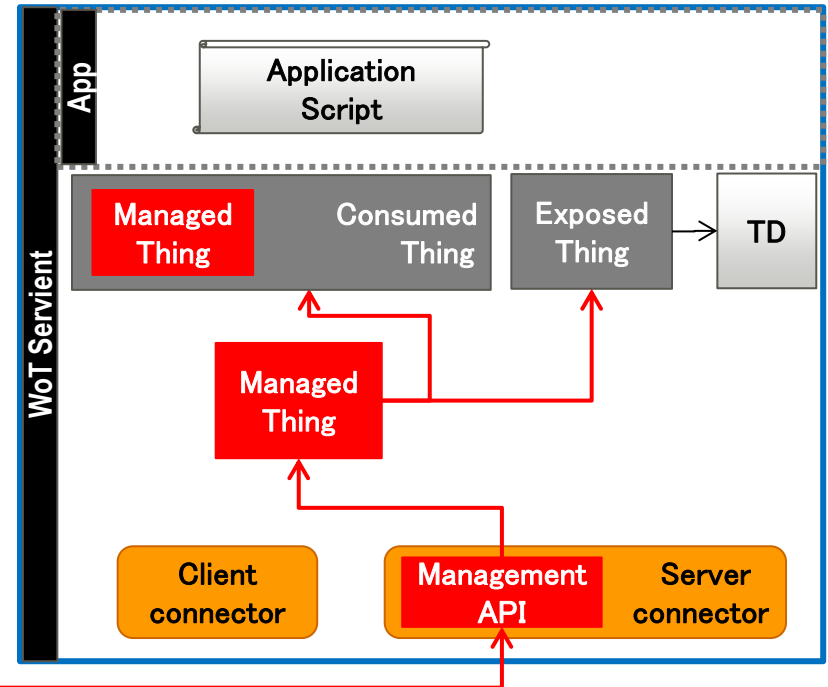
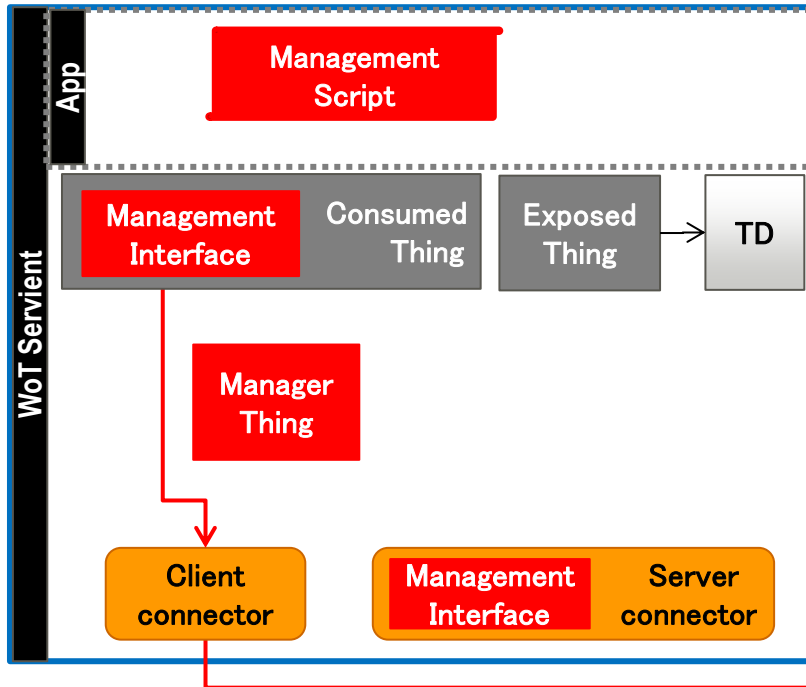
# Management Interface

- ManagedThing executes the management commands from Manager interface in Client.



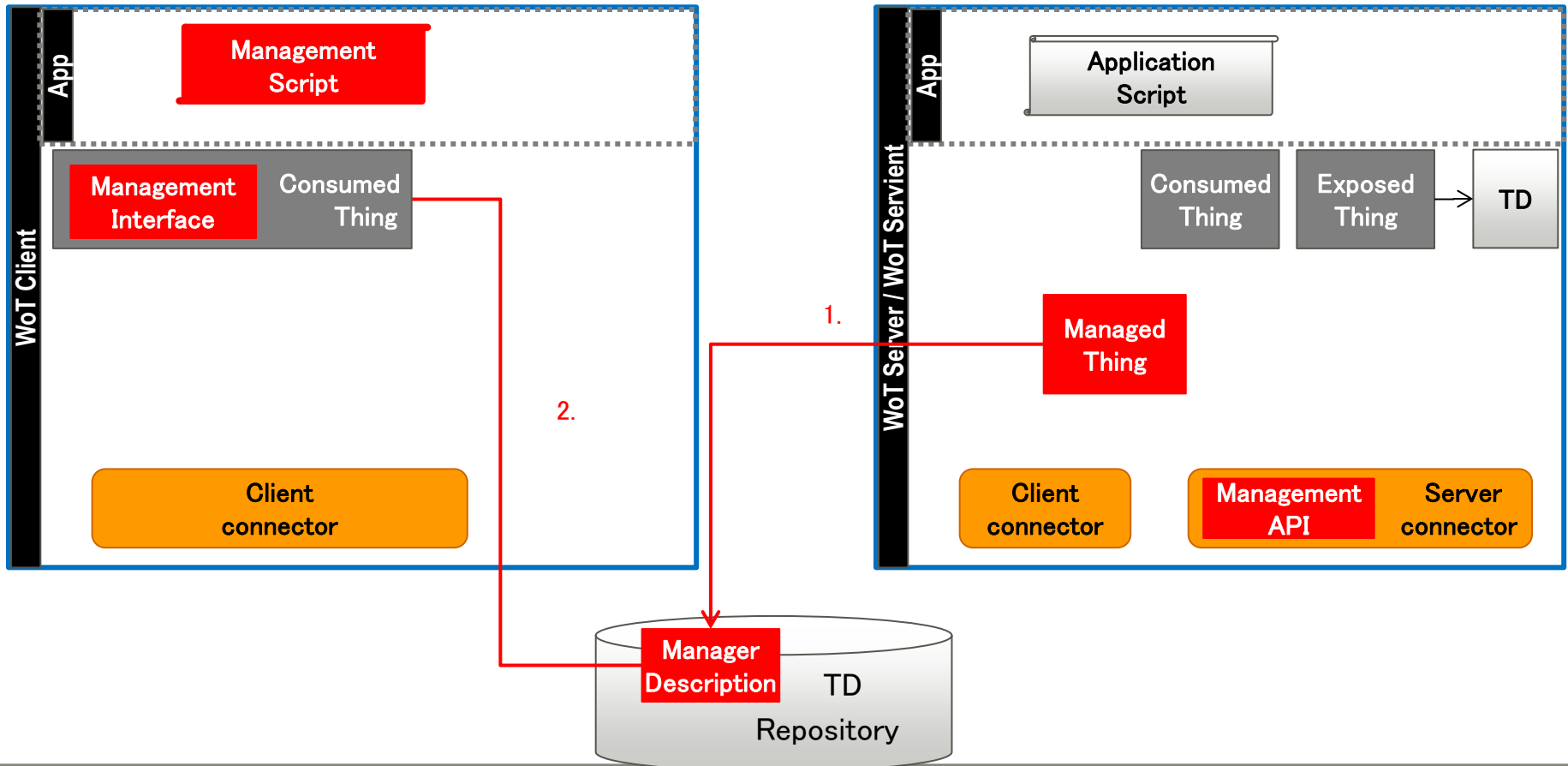
# Management Interface

- ManagedThing can communicate between WoT Servients.



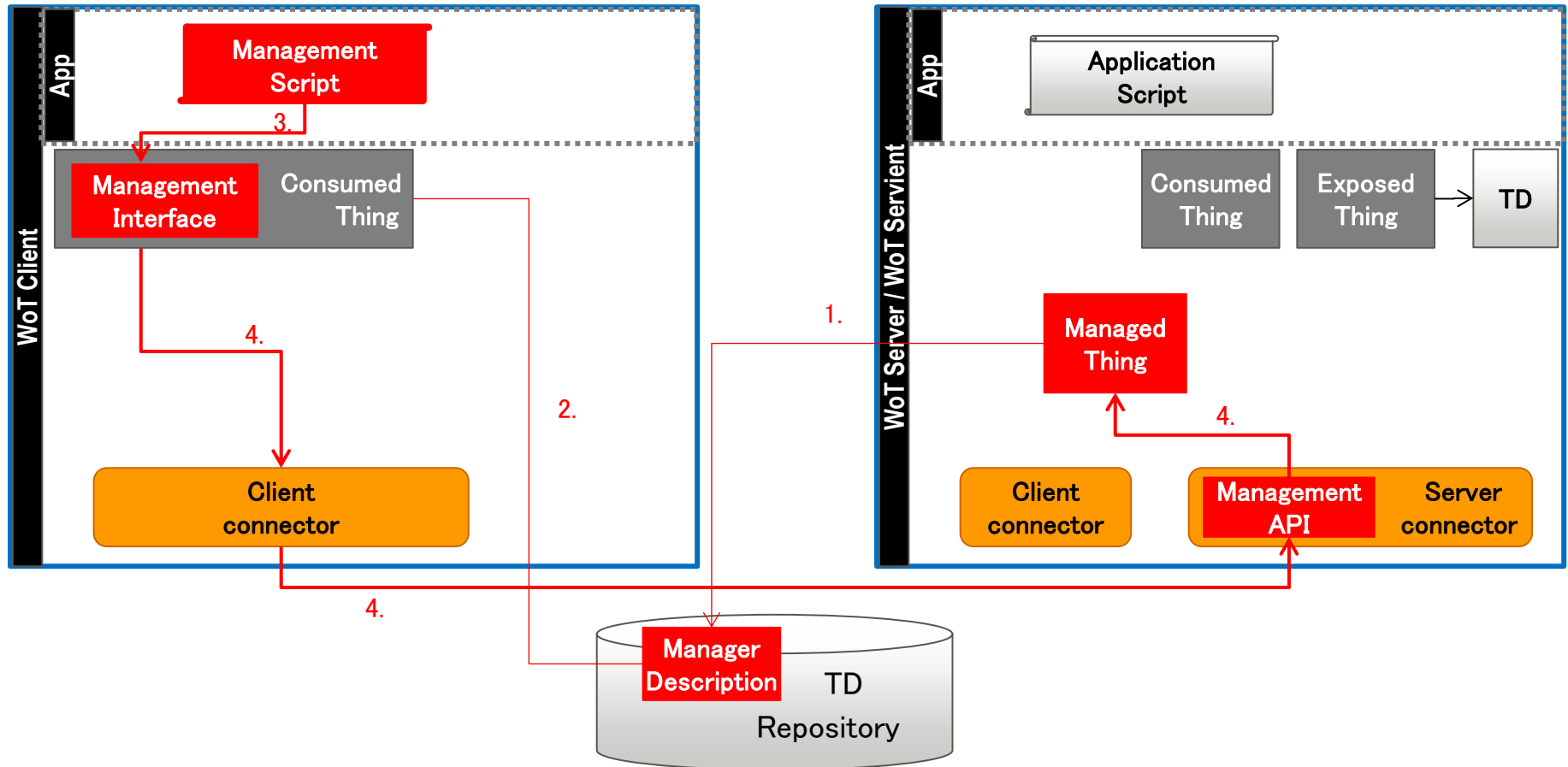
# Behavior of management Interface

1. ManagerThing registers ManagerDescription describing the supported management interface.
2. Management Interface in ConsumedThing downloads the ManagerDescription and prepares the interface.



# Behavior of management Interface

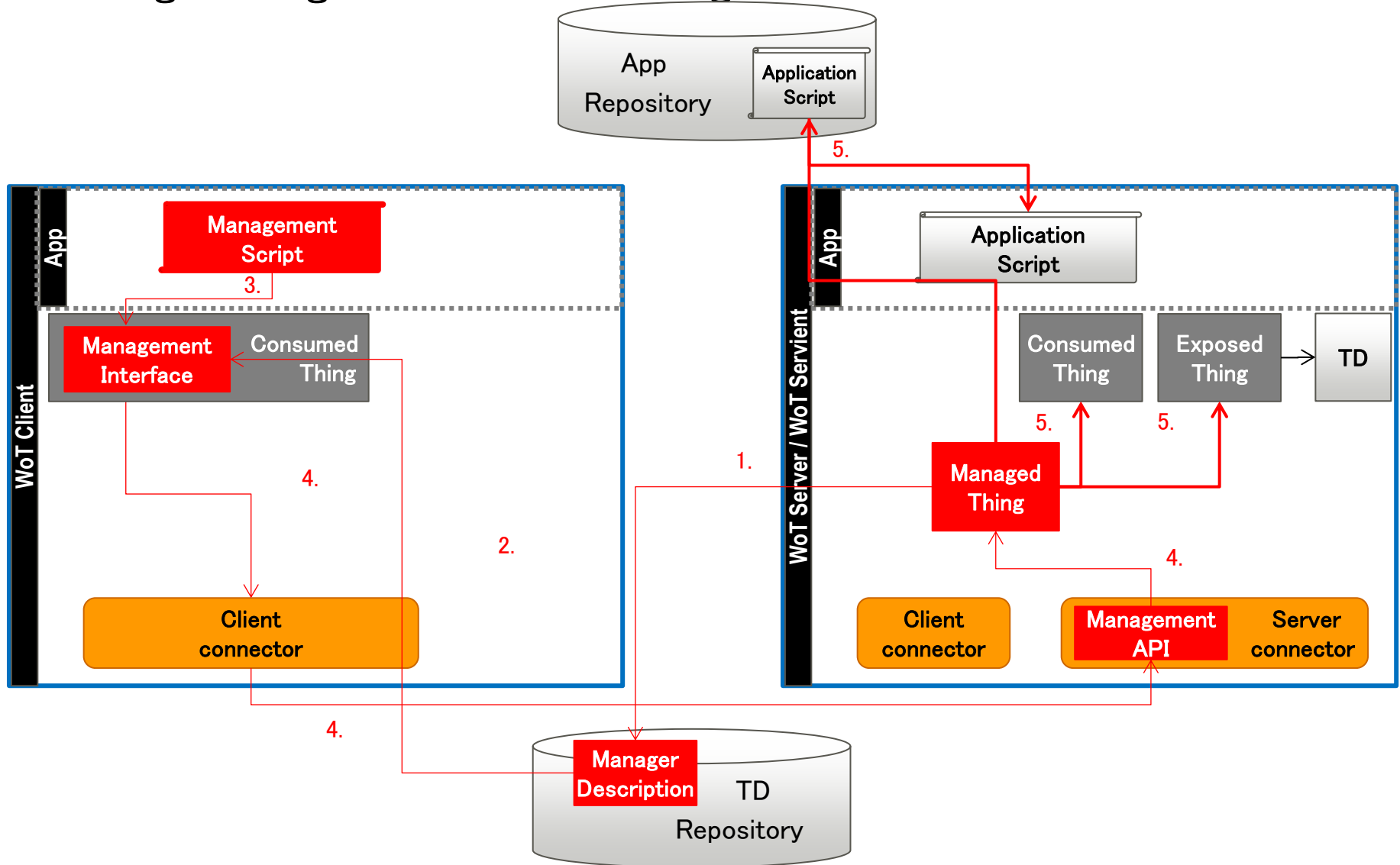
- 3. Management Interface interprets management script.
- 4. Management Interface sends management command to Management API → ManagedThing.



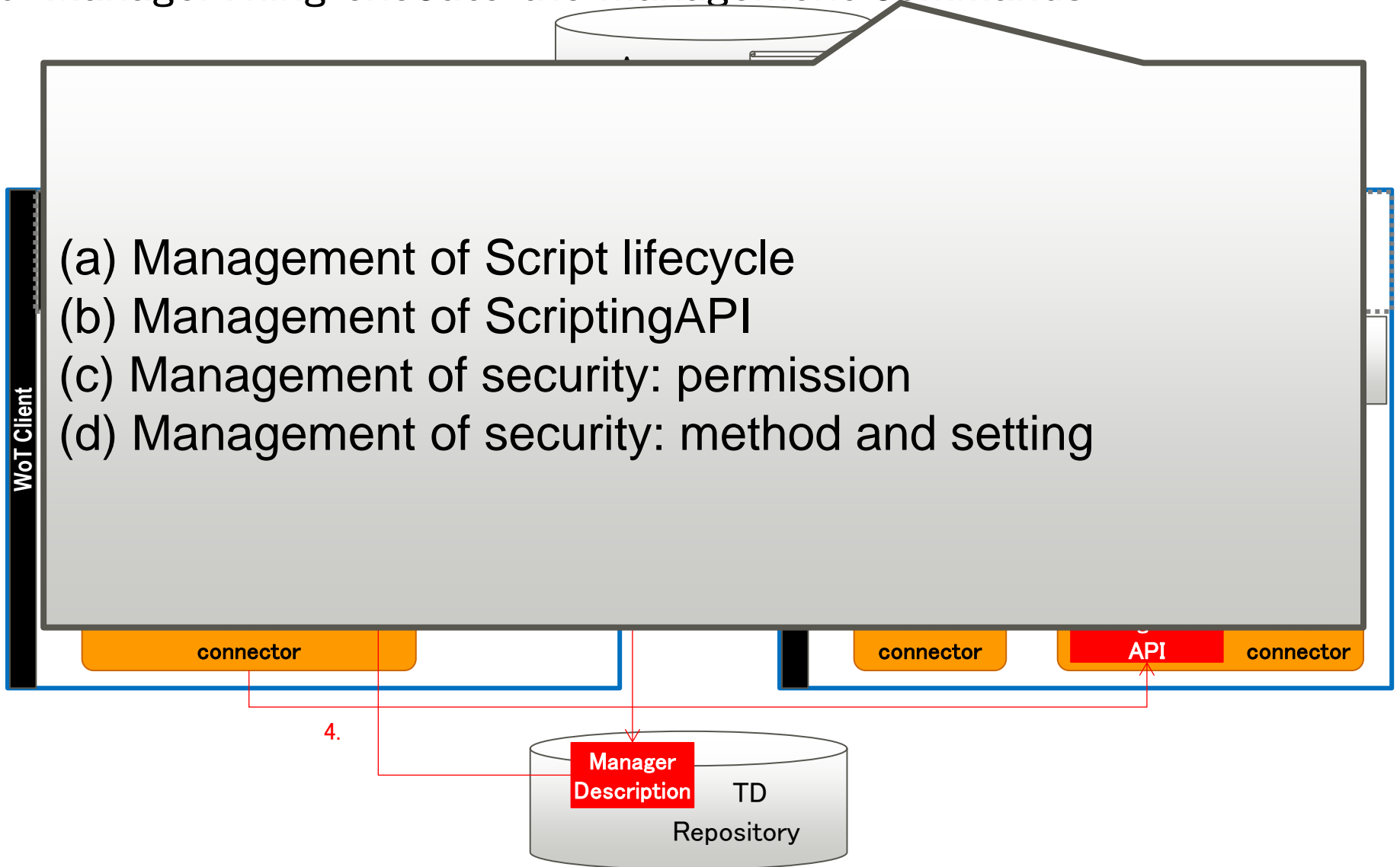


# Behavior of management Interface

## 5. ManagerThing execute the management commands.



## 5. ManagerThing execute the management commands.

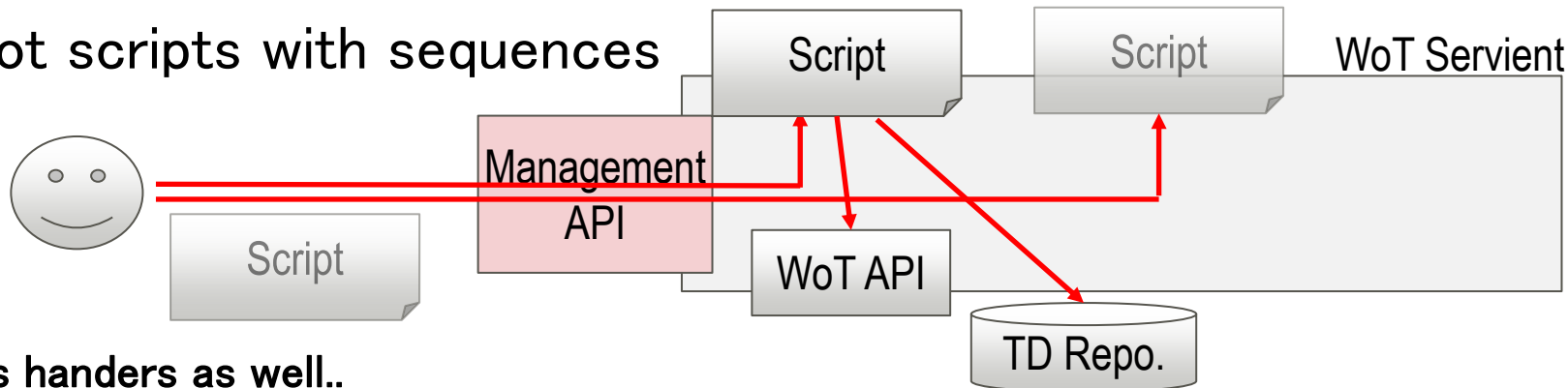


# (a) Management of Script lifecycle

- Register a script remotely.
  - A script is registered and interpreted, then exposes WoT API and registers TD.
- Register another script remotely.
  - Another script is added/injected to the executing script, then another WoT API is exposed and TD is updated.
- Unregister a script / all scripts remotely.
  - A script / all scripts are unregistered, then closes the WoT APIs and unregisters the TDs.

## ■ Start/Stop scripts

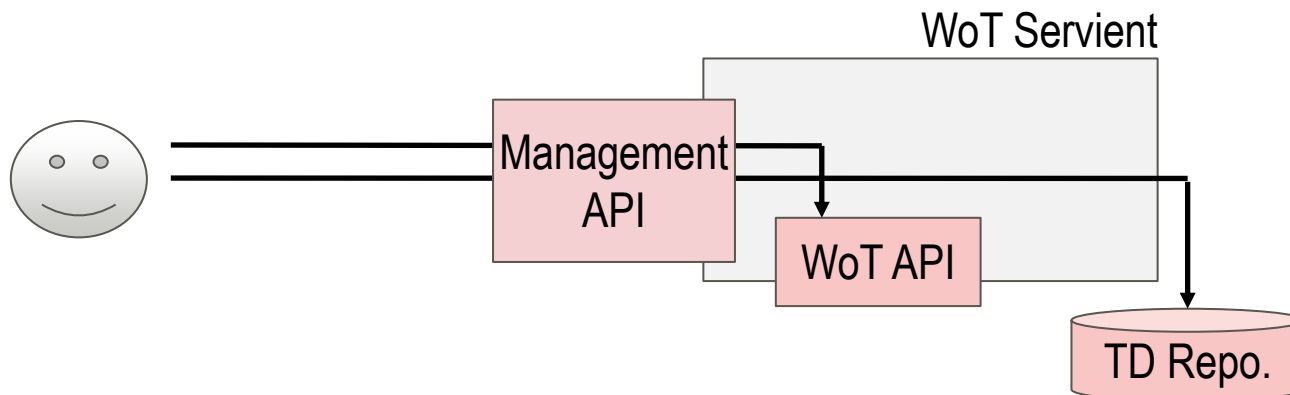
## ■ (Re)boot scripts with sequences



Note: It deals handlers as well..

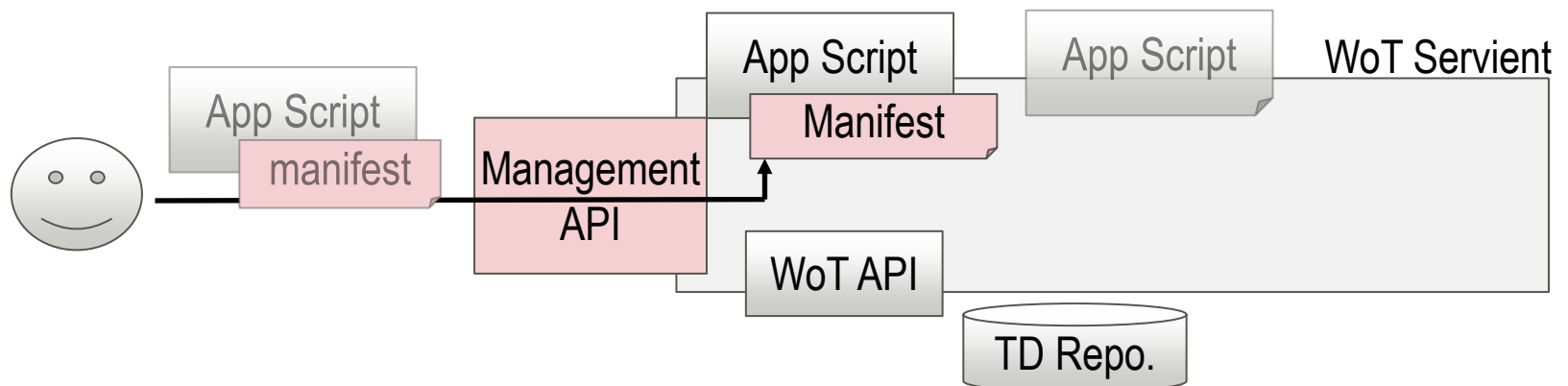
## (b) Management of ScriptingAPI

- Open/close WoT API remotely without script execution on WoT Servient.
  - WoT API that a script exposed via ExposedThing can be closed/opened remotely.
- Register/unregister TD remotely without script execution on WoT Servient.
  - TD that a script registered via ExposedThing can be registered/unregistered remotely.



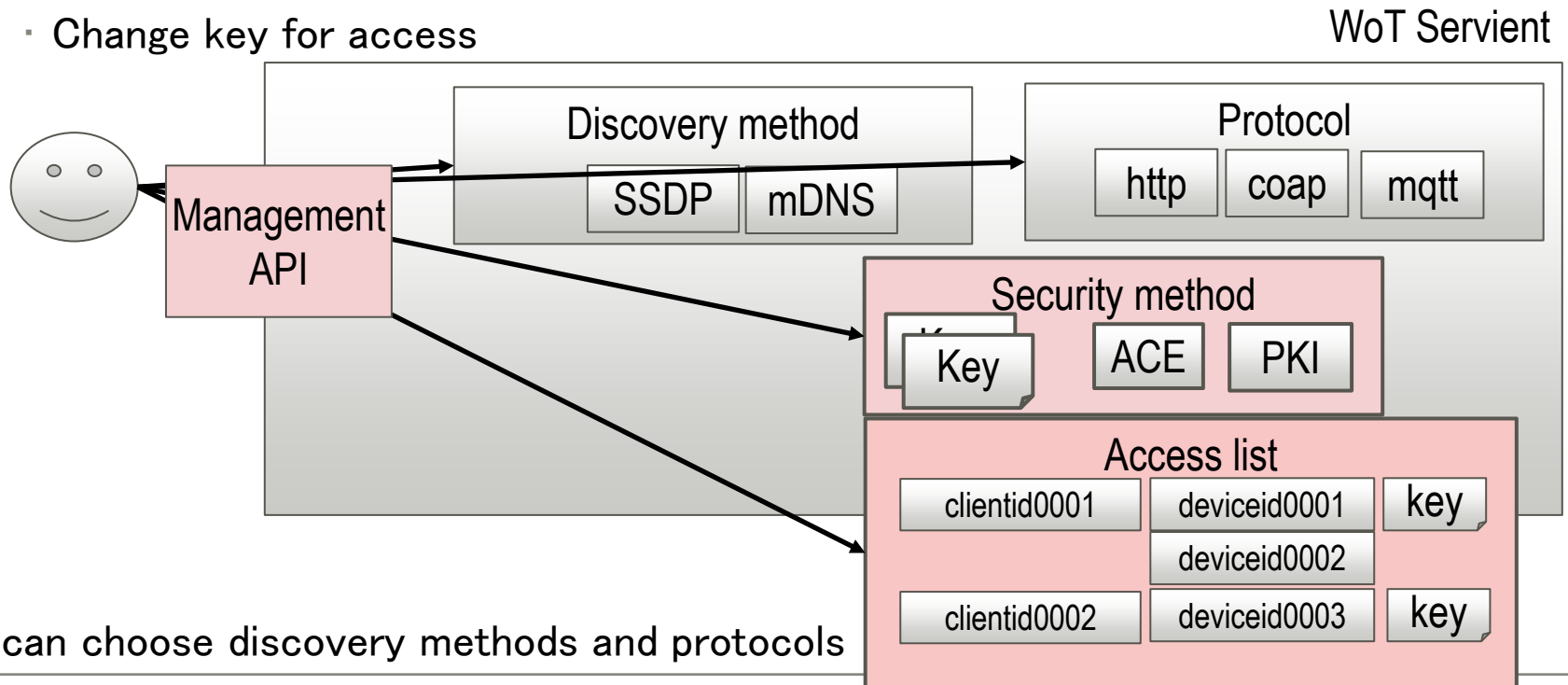
## (c) Management of security: permission

- Permission can be set through management interface.
- Confirm right to use management interface and give the permission
  - Runtime level: When a script is registered, confirm whether the runtime is allowed to use the interface.
  - Script level: When a script is registered, a manifest is checked whether the script can be executed on the Servient.
- Prevent unauthorized access by restricting APIs that scripts can use
  - API level: The script can call APIs only when those are permitted i.e. specified in Manifest.



# (d) Management of security: method and setting

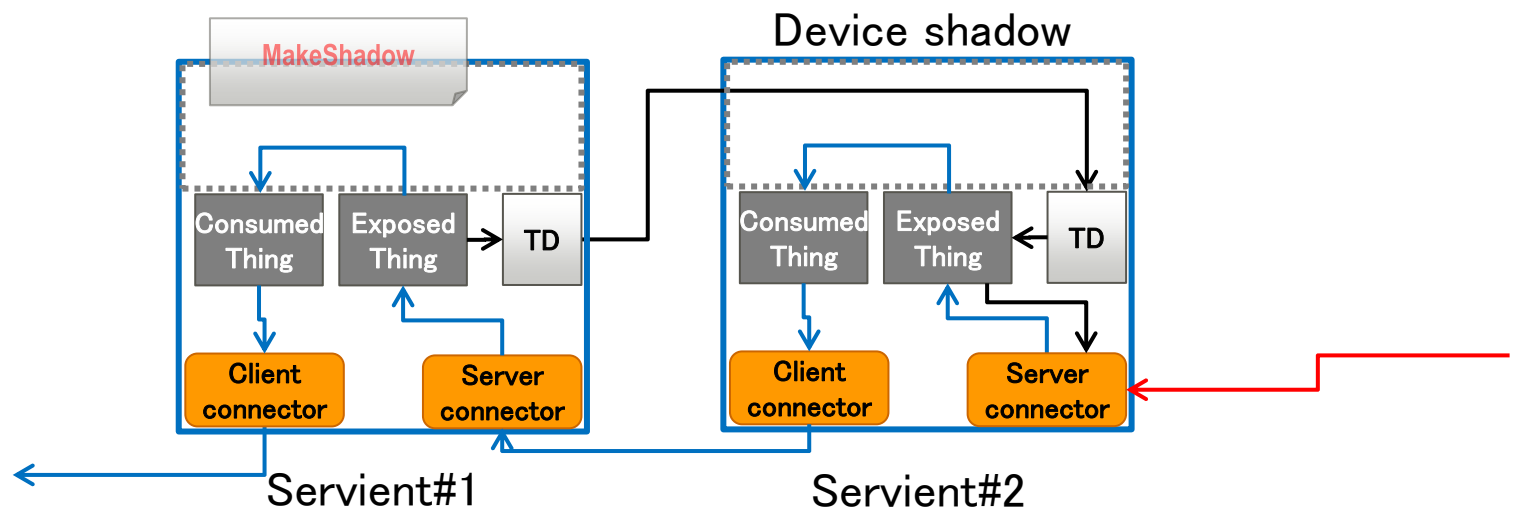
- Choose methods of security remotely.
  - Security method used by WoT Servient can be changed.
- After choosing a method, manages the security setting remotely.
  - E.g. Control permission
    - Authorize clients to access devices (ConsumedThing → Device)
    - Cancel authorization for clients
    - Change key for access




\* Also can choose discovery methods and protocols

# How to achieve the Synchronization

- Synchronization can be realized as an application of Scripting API. Not necessary to use management interface.
  - Set Servient#1 to generate TD and Servient#2 to generate API from TD.
    - If TD changed, e.g. IP address change, it is reflected to the TD. Servient #2 is notified the change.
    - If TD changed, e.g. adding a new capability like gradually turn off the Lamp, it is reflected to the TD. Servient #2 is notified the change and reflected to the API exposure.
  - Connect ConsumedThing #2 and ExposedThing #1 in Servient #2.
    - When WoT API of Servient #2 is called, the callback function calls ExposedThing #1 .
  - Device shadow can have application script working between server call and client call in Servient #2.





**FUJITSU**

shaping tomorrow with you