Sertainty™

Data Protector Web API

Guide

Version: V3.4.0

Table of Contents

1. Overview

The Sertainty HTTP service is a proprietary system that enables communication with the Workflow host gateway. The host performs various functions, including:

- UXP message delivery
- UXP license validation and management
- Workflow ECO-System
- UXP management
- Scatter management

2. Request and Response Format

The format of a request is:

```
<Request>
  <Session>session-id</Session>
  <Function>function-name</Function>
  <ArgList>
    <Argument>base64-data</Argument>
    <Argument>base64-data</Argument>
    …
  </ArgList>
</Request>
```

Table 1 – Request Elements

Element	Description
<Request>	The outer most XML tag for the request document. **Required**
<Session>	Specifies the required session identifier as assign at API authentication time.
<Function>	Name of the requested function. **Required**
<ArgList>	Indicates a list of one or more arguments are to be passed to the function. If no arguments are required by the function, then tag is optional. **Required for arguments**
<Argument name="n">	An argument to be passed to the function. The data must be in Base64 format. The attribute **name** is used to identify an argument and may be optional. **Required for arguments**

The format of a response is:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>base64-data</Result>
    <Result>base64-data</Result>
    <Result>base64-data</Result>
    …
  </ResultList>
</Response>
```

Table 2 – Response Elements

Element	Description
<Response>	The outer most XML tag for the response document. **Required**
<Status>	Status code indicating the result of the function call. **Required**
<StatusMessage>	Static message that is associated with the **<Status>** element. **Required**
<ResultList>	Indicates a list of one or more elements have been returned from the function. If no elements are returned by the function, then tag is optional. **Required for result data**
<Result name="n">	A return data element from the function. The data must be in Base64 format. The attribute **name** is used to identify the result item and may be optional. **Required for result data**

3. Service API

The HTTP service API is as follows: TBD

4. Workflow Functions

There will only be one API function that can communicate with a server; however, the single API is overloaded to perform unlimited logical functions. Response values are also overloaded.

A request and its corresponding response are in XML format.

Table 3 – Logical Functions

Function	*Description*
agentAlter	Alters the agent status.
agentGetLog	Gets the agent log.
agentGetStatus	Gets the current active configuration and the runtime status.
agentPing	Pings the agent.
agentRestart	Restarts the agent.
agentShutdown	Stops the agent.
agentStart	Starts the agent.
configActivate	Activates a configuration for execution.
configCreate	Creates a new configuration.
configDelete	Deletes a configuration.
configGetID	Gets an existing ID from the configuration.
configGetList	Gets a list of configuration names.
configGetScript	Gets an existing script from the configuration.
configGetStatus	Gets the configuration runtime status.
configGetTasks	Gets the tasks for a configuration.
configNewID	Creates a new machine ID.
configSaveID	Saves an ID to the configuration.
configSaveScript	Saves a script to the configuration.
configSaveTasks	Saves the tasks to a configuration.
hostCloseSession	Closes a previously open session.
hostGetMachine	Gets the machine fingerprint of the agent host.
hostOpenSession	Requests access to the Sertainty gateway.
sharedGetID	Gets an existing ID shared by all configurations.
sharedGetPreset	Gets a rule preset.
sharedGetPresetList	Gets the list of rule preset names.
sharedGetScript	Gets an existing script shared by all configurations.
sharedNewID	Creates a new machine ID shared by all configurations.
sharedSaveID	Saves an ID shared by all configurations.
sharedSaveScript	Saves a script shared by all configurations.
taskGetLog	Gets the log for a task.
taskGetStatus	Gets the status record for a task.
taskPause	Pauses a task.
taskReset	Resets a task execution statistics and log data.
taskResume	Resumes a paused task.
templateGetList	Gets a list of template names.
templateGetTask	Gets a template task definition.
templateSaveTask	Saves a task definition as a template.

4.1. agentAlter

Alters the agent runtime status.

Format:

```
<Request>
  <Function>agentAlter</Function>
  <ArgList>
    <Argument>session-id</Argument>
  </ArgList>
</Request>
```

Function Arguments:

session-id	Specifies the assigned session identifier.
alter-option	Specifies the option value to set. Possible values: • Disable – Disables the agent workflow system. • Enable – Enables the agent workflow system.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.2. agentGetLog

Gets the master log for the agent. Every time the agent starts, a new log is created using a time-based naming convention. The master log is in XML format.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>agentGetLog</Function>
</Request>
```

Function Arguments:

None

Returns:

clear-log-data	Contains entire activity log for the agent.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>clear-log-data</Result>
  </ResultList>
</Response>
```

4.3. agentGetStatus

Gets the activate configuration name and the runtime status of the system.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>agentGetStatus</Function>
</Request>
```

Function Arguments:

None

Returns:

agent-status	Status of the agent. Possible values: • Active - system is active and running. • Disabled – system has been temporarily disabled. • Offline – system offline or inaccessible.
application-name	Name of the application that manages the system.
application-version	Version of the application that manages the system.
config-name	Specifies the name of the active configuration.
config-status	Status of the active configuration. Possible values: • Active – configuration can run tasks. • Disabled – configuration has been temporarily disabled.

	• Idle – configuration is idle due to agent being offline.
task-status	An aggregate status of configuration tasks. Possible values: • Errors – one or more tasks contain an error. • Normal – all scheduled tasks have no errors.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>agent-status</Result>
    <Result>application-name</Result>
    <Result>application-version</Result>
    <Result>config-name</Result>
    <Result>config-status</Result>
    <Result>task-status</Result>
  </ResultList>
</Response>
```

4.4. agentPing

Pings the requested agent. A timeout may indicate an unreachable agent or an agent that is not currently running.

Format:

```
<Request>
  <Function>agentPing</Function>
</Request>
```

Function Arguments:

None

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.5. agentRestart

Restarts the agent. A restart will terminate all scheduled tasks and reset the runtime log.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>agentRestart</Function>
</Request>
```

Function Arguments:

None

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.6. agentShutdown

Stops the agent. A shutdown will terminate all running and scheduled tasks. **Note**: to start the agent, a system administrator must connect to the host and manually start the agent.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>agentShutdown</Function>
</Request>
```

Function Arguments:

None

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.7. agentStart

Starts the agent.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>agentStart</Function>
</Request>
```

Function Arguments:

None

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.8. configActivate

Activates a configuration. A configuration must be active for tasks to automatically execute.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configActivate</Function>
  <ArgList>
    <Argument>config-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration to activate.

Returns:

```
<Response>
  <Status>error-code</Status>
```

```
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.9. configCreate

Creates a new configuration. This will set up the configuration on the host machine with an empty task list.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configCreate</Function>
  <ArgList>
    <Argument>config-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the new configuration. The configuration must not exist.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.10. configDelete

Deletes an existing configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configDelete</Function>
  <ArgList>
    <Argument>config-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configruation to delete. It will delete all existing task data, including logs and status data.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
 </Response>
```

4.11. configGetID

Gets the specified ID from the configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configGetID</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>id-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration containing ID.
id-name	Specifies the ID filename to retrieve. The name must contain the full file name.

Returns:

id-contents	Specifies the binary contents of the requested ID.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
```

```
    <Result>id-contents</Result>
  </ResultList>
 </Response>
```

4.12. configGetList

Gets a list of all configurations.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configGetList</Function>
</Request>
```

Function Arguments:

None

Returns:

config-name	Specifies the name of a valid configuration.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>config-name</Result>
    <Result>config-name</Result>
    ...
  </ResultList>
 </Response>
```

4.13. configGetScript

Gets the specified script from the configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configGetScript</Function>
```

```
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>script-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration containing ID.
script-name	Specifies the script filename to retrieve. The name must contain the full file name.

Returns:

script-contents	Specifies the contents of the requested script.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>script-contents</Result>
  </ResultList>
 </Response>
```

4.14. configGetStatus

Gets the runtime status of the configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configGetStatus</Function>
  <ArgList>
    <Argument>config-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configruation for which status will be returned.

Returns:

config-status	Status of the active configuration. Possible values: • Active – configuration can run tasks. • Disabled – configuration has been temporarily disabled. • Idle – configuration is idle due to agent being offline.
task-status	An aggregate status of configuration tasks. Possible values: • Errors – one or more tasks contain an error. • Normal – all scheduled tasks have no errors.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>config-status</Result>
    <Result>task-status</Result>
  </ResultList>
</Response>
```

4.15. configGetTasks

Get the task definitions for the configuration. Tasks are defined as an XML document.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configGetTasks</Function>
  <ArgList>
    <Argument>config-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configruation for which tasks will be returned.

Returns:

task-definitions	Specifies the contents of the task definitions in XML format.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>task-definitions</Result>
  </ResultList>
 </Response>
```

4.16. configNewID

Create the specified ID in the configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configNewID</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>id-name</Argument>
    <Argument>machine-fingerprint</Argument>
    <Argument>machine-fingerprint</Argument>
    …
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration to receive the ID.
id-name	Specifies the ID filename to create. The name must contain the full file name.
machine-fingerprint	Specifies a machine fingerprint to add to the ID. If no fingerprints are provided, the ID will contain the fingerprint of the current machine. If one or more fingerprints are provided, the ID will only contain the specified fingerprints.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.17. configSaveID

Saves the specified ID in the configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configSaveID</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>id-name</Argument>
    <Argument>id-contents</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration to receive the ID.
id-name	Specifies the ID filename to save. The name must contain the full file name.
id-contents	Specifies the ID contents.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.18. configSaveScript

Saved the specified script in the configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configSaveScript</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>script-name</Argument>
    <Argument>script-contents</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration to receive the script.
script-name	Specifies the script filename to save. The name must contain the full file name.
script-contents	Specifies the script contents.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.19. configSaveTasks

Save the task definitions for the configuration. Tasks are defined as an XML document.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>configSaveTasks</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>tasks-contents</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the 24onfiguration for which tasks will be saved.
tasks-contents	Specifies tasks XML document that contains all task definitions.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.20. hostCloseSession

Closes a previously open session.

Format:

```
<Request>
  <Function>hostCloseSession</Function>
  <ArgList>
    <Argument>session-id</Argument>
  </ArgList>
</Request>
```

Function Arguments:

session-id	Specifies the assigned session identifier.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.21. hostGetMachine

Gets the host machine fingerprint. The fingerprint can then be used to create a machine ID.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>hostGetMachine</Function>
</Request>
```

Function Arguments:

None

Returns:

fingerprint	Specifies the host machine fingerprint in XML format.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>fingerprint</Result>
  </ResultList>
</Response>
```

4.22. hostOpenSession

Requests access to Sertainty gateway. To authenticate, the caller must call the function in a loop until access is granted, denied or aborted. When a session is granted, the caller must use the returned **session-id** for future calls.

Format:

```
<Request>
  <Function>hostOpenSession</Function>
  <ArgList>
    <Argument>client-user</Argument>
    <Argument name="prompt-n">prompt-response</Argument>
    <Argument name="prompt-n">prompt-response</Argument>
  </ArgList>
</Request>
```

Function Arguments:

client-user	Specifies the active client username.

prompt-n	Specifies the requested prompt value. Depending upon trust assessment, there may be zero to **N** prompts that require a response.

Returns:

session-id	Specifies the assigned session identifier. The session identifier is used for all other function calls.
prompt-n	Specifies a prompt that must be answered by the caller.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>session-id</Result>
  </ResultList>
</Response>

<Response>
  <Status>3</Status>
  <StatusMessage>Denied</StatusMessage>
</Response>

<Response>
  <Status>2</Status>
  <StatusMessage>Challenged</StatusMessage>
  <ResultList>
    <Result>prompt-1</Result>
    <Result>prompt-2</Result>
    <Result>prompt-3</Result>
    …
  </ResultList>
</Response>
```

4.23. sharedGetID

Gets the specified ID from the shared area. The ID can be accessed by all configurations.

Format:

```
<Request>
  <Session>session-id</Session>
```

```
  <Function>sharedGetID</Function>
  <ArgList>
    <Argument>id-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

id-name	Specifies the ID filename to retrieve. The name must contain the full file name.

Returns:

id-contents	Specifies the binary contents of the requested ID.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>id-contents</Result>
  </ResultList>
 </Response>
```

4.24. sharedGetScript

Gets the specified script from the shared area. The script can be accessed by all configurations.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>sharedGetScript</Function>
  <ArgList>
    <Argument>script-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

script-name	Specifies the script filename to retrieve. The name must contain the full file name.

Returns:

script-contents	Specifies the contents of the requested script.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>script-contents</Result>
  </ResultList>
 </Response>
```

4.25. sharedGetPreset

Gets the specified rule preset from the host. The preset can be applied to an ID definition.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>sharedGetPreset</Function>
  <ArgList>
    <Argument>preset-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

prset-name	Specifies the rule preset to retrieve.

Returns:

preset-contents	Specifies the rule preset in XML format.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>preset-contents</Result>
  </ResultList>
 </Response>
```

4.26. sharedGetPresetList

Gets a list of rule presets from the host.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>sharedGetPresetList</Function>
</Request>
```

Function Arguments:

None

Returns:

preset-name	Specifies the rule preset name.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>preset-name1</Result>
    <Result>preset-name2</Result>
    <Result>preset-name3</Result>
    <Result>preset-name4</Result>
  </ResultList>
 </Response>
```

4.27. sharedNewID

Create the specified ID in a shared area. The ID can be accessed by all configurations.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>sharedNewID</Function>
  <ArgList>
    <Argument>id-name</Argument>
```

```
    <Argument>machine-fingerprint</Argument>
    <Argument>machine-fingerprint</Argument>
    …
  </ArgList>
</Request>
```

Function Arguments:

id-name	Specifies the ID filename to create. The name must contain the full file name.
machine-fingerprint	Specifies a machine fingerprint to add to the ID. If no fingerprints are provided, the ID will contain the fingerprint of the current machine. If one or more fingerprints are provided, the ID will only contain the specified fingerprints.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.28. sharedSaveID

Saves the specified ID in a shared area. The ID can be accessed by all configurations.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>sharedSaveID</Function>
  <ArgList>
    <Argument>id-name</Argument>
    <Argument>id-contents</Argument>
  </ArgList>
</Request>
```

Function Arguments:

id-name	Specifies the ID filename to save. The name must contain the full file name.

id-contents	Specifies the ID contents.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.29. sharedSaveScript

Saved the specified script in a shared area. The script can be accessed by all configurations.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>sharedSaveScript</Function>
  <ArgList>
    <Argument>script-name</Argument>
    <Argument>script-contents</Argument>
  </ArgList>
</Request>
```

Function Arguments:

script-name	Specifies the script filename to save. The name must contain the full file name.
script-contents	Specifies the script contents.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.30. taskGetLog

Get the task log. The log is in XML format.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>taskGetLog</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>task-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configruation containing the task.
task-name	Specifies the name of task.

Returns:

task-log	Contains the contents of the request task log in XML format.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>task-log</Result>
  </ResultList>
 </Response>
```

4.31. taskGetStatus

Get the task status data.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>taskGetStatus</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>task-name</Argument>
  </ArgList>
```

```
</Request>
```

Function Arguments:

config-name	Specifies the name of the configruation containing the task.
task-name	Specifies the name of task.

Returns:

StartTime	Date and time of last execution.
EndTime	Date and time of last execution completion.
LastErrorTime	Date and time of last task error.
Status	Current status of task. Possible values: • Error – Most recent exeution failed. • Executing – Task is currently running. • Idle – Task exeution is disabled. • Paused – Task execution has been paused. • Waiting – Task is waiting for next scheduled execution.
Error	Error message for last task error.
ExecCnt	Total number of task executions.
ErrorCnt	Total number of task errors.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result name="StartTime">start-time</Result>
    <Result name="EndTime">end-time</Result>
    <Result name="LastErrorTime">error-time</Result>
    <Result name="Status">status-value</Result>
    <Result name="Error">error-message</Result>
    <Result name="ExecCnt">exec-cnt</Result>
    <Result name="ErrorCnt">error-cnt</Result>
  </ResultList>
 </Response>
```

4.32. taskPause

Pauses task execution.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>taskPause</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>task-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configruation containing the task.
task-name	Specifies the name of task. If an asterisk is specified, all tasks will be paused.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.33. taskReset

Resets the runtime data, including log info, for a task within a configuration.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>taskReset</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>task-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration for which tasks will be reset.
task-name	Specifies the name of task. If an asterisk is specified, all tasks will be reset.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.34. taskResume

Resume paused task execution.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>taskResume</Function>
  <ArgList>
    <Argument>config-name</Argument>
    <Argument>task-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

config-name	Specifies the name of the configuration containing the task.
task-name	Specifies the name of task. If an asterisk is specified, all tasks will resume.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

4.35. templateGetList

Gets a list of all templates.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>templateGetList</Function>
</Request>
```

Function Arguments:

None

Returns:

task-name	Specifies the name of a valid template task.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>task-name</Result>
    <Result>task-name</Result>
    ...
  </ResultList>
 </Response>
```

4.36. tempateGetTask

Get the task definition for the template. Tasks are defined as an XML document.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>templateGetTask</Function>
  <ArgList>
    <Argument>task-name</Argument>
  </ArgList>
</Request>
```

Function Arguments:

task-name	Specifies the name of the template task to be returned.

Returns:

task-definition	Specifies the contents of the task definition in XML format.

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
  <ResultList>
    <Result>task-definition</Result>
  </ResultList>
 </Response>
```

4.37. templateSaveTask

Saves the specified task as a template task.

Format:

```
<Request>
  <Session>session-id</Session>
  <Function>templateSaveTask</Function>
  <ArgList>
    <Argument>task-definition</Argument>
  </ArgList>
</Request>
```

Function Arguments:

task-definition	Specifies the task definition in XML format.

Returns:

```
<Response>
  <Status>error-code</Status>
  <StatusMessage>error-message</StatusMessage>
</Response>
```

5. Error Codes

The following table lists possible error codes returned by the HTTP API:

Table 4 – Error Codes

Code	Description
1	The requested operation was successful.
2	The user must respond to authentication challenges.
3	Access has been denied.
213	The requested file was not found.
232	The specified file already exists.
408	The specified request is invalid.
502	A general error has occurred. The corresponding text will describe the general error.