FUJITSU

FUJITSU Software Systemwalker Software Configuration Manager

Developer's Guide

Windows/Linux

B1X1-0312-02ENZ0(00)
August 2014

Preface

Purpose of this Document

This document explains how to independently develop the following definitions for managing software parameters using Systemwalker Software Configuration Manager V15.3.0.

- Software information definitions
- Parameter information definitions

Intended Readers

This document is intended for those engaged in developing parameter management definitions for Systemwalker Software Configuration Manager.

It is assumed that readers of this document already have the following knowledge:

- Expertise in the software to be managed
- Expertise in programming

Structure of this Document

The structure of this document is as follows:

Conventions Used in this Document

Refer to the *Documentation Road Map* for information on the names, abbreviations, and symbols used in this manual.

Abbreviations and Generic Terms Used for Operating Systems

This document uses the following abbreviations and generic terms to indicate operating systems.

Official name	Abbreviation	
Microsoft(R) Windows Server(R) 2012 Datacenter Microsoft(R) Windows Server(R) 2012 Standard	Windows Server 2012	Windows
Microsoft(R) Windows Server(R) 2012 R2 Datacenter Microsoft(R) Windows Server(R) 2012 R2 Standard	Windows Server 2012 R2	
Microsoft(R) Windows Server(R) 2008 Standard Microsoft(R) Windows Server(R) 2008 Standard without Hyper-V	Windows Server 2008	

Official name	Abbreviation	
Microsoft(R) Windows Server(R) 2008 Enterprise Microsoft(R) Windows Server(R) 2008 Enterprise without Hyper-V		
Microsoft(R) Windows Server(R) 2008 R2 Standard Microsoft(R) Windows Server(R) 2008 R2 Enterprise	Windows Server 2008 R2	
Microsoft(R) Windows Server(R) 2003 R2, Standard Edition Microsoft(R) Windows Server(R) 2003 R2, Enterprise Edition Microsoft(R) Windows Server(R) 2003 R2, Standard x64 Edition Microsoft(R) Windows Server(R) 2003 R2, Enterprise x64 Edition	Windows Server 2003 R2	
Red Hat(R) Enterprise Linux(R) (for x86)	RHEL (x86)	RHEL
Red Hat(R) Enterprise Linux(R) (for Intel64)	RHEL (Intel64)	
Oracle Solaris	Solaris Operating System Solaris OS	Solaris

Export Restrictions

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

Trademarks

- Adobe, Adobe Reader, and Flash are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.
- Interstage, ServerView, and Systemwalker are registered trademarks of Fujitsu Limited.
- Linux is a registered trademark of Linus Torvalds.
- Red Hat, RPM, and all Red Hat-based trademarks and logos are trademarks or registered trademarks of Red Hat, Inc. in the United States and other countries.
- Oracle and Java are registered trademarks of Oracle and/or its affiliates in the United States and other countries. Company names and product names used in this document are registered trademarks or trademarks of those companies.
- VMware, the VMware "boxes" logo and design, Virtual SMP, and VMotion are registered trademarks or trademarks of VMware, Inc. in the United States and/or other jurisdictions.
- Xen and XenSource are trademarks or registered trademarks of XenSource, Inc. in the United States and/or other countries.
- Other company names and product names are trademarks or registered trademarks of their respective owners.

Note that system names and product names in this document are not accompanied by trademark symbols such as (TM) or (R).

Issue Date and Version

Version	Manual code
March 2014: First edition	B1X1-0312-01ENZ0(00) / B1X1-0312-01ENZ2(00)
August 2014: Second edition	B1X1-0312-02ENZ0(00) / B1X1-0312-02ENZ2(00)

Copyright

Contents

Chapter 1 Definition of Software Information

The following definitions are required to manage software that is not compatible with UpdateAdvisor (middleware) as installed software, or to manage the software parameters:

- Definition of Software Information
- Definition of Installed Software

1.1 Definition of Software Information

Use swcfmg_software (Software Information Management Command) to define software information and perform the operations below (refer to "Chapter 3 Command Reference" for information on the commands):

- Registration of software information
- Update of software information
- Deletion of software information
- Output of software information

Registration of software information

Register the basic software information in the CMDB. Note that a "software ID" number will be assigned automatically during registration.

[Registration information]

- Software ID <*Software ID*>
- Software name <*Software name*>
- Version <*Version*>
- OS type <*OS type*>
- Vendor <*Vendor*>

Use "<*Software name*> + <*Version*> + <*OS type*>" to register unique information. Software with identical information is deemed to be the same, and cannot be registered in the CMDB.

To register software information, specify the CSV files in the format below using the Software Information Management Command (refer to "Chapter 4 File Reference" for information on file format):

```
[#][<Software ID>],<Software name>,<Version>,<OS type>,<Vendor>
...
```

Notes for new registrations

<*Software ID*> cannot be entered for a new registration. A registration error will occur if this happens.

Update of software information

Update the registered software information.

To update software information, specify CSV files in the format above using the Software Information Management Command.

The "software ID" automatically assigned during registration is used as the key when updating. Use the software information output feature to check the ID of the registered software information.

When updating information, we recommend revising files generated using the software information output feature, and deleting software information that does not require update. Use these files as the input for the command to update information.

Note that an update error will occur if no software ID, or an incorrect one, is entered.

Deletion of software information

Delete the registered software information.

Specify the software ID of the software information to be deleted.

Check the software ID using the software information output function.

Note that software information cannot be deleted for software that has registered installed software information. Ensure that installed software information has been deleted before deleting the software information that is not required.

Output of software information

Output the registered software information in the CSV format below.

The output destination is the standard output of the command or the specified file.

```
#Software ID,Software name,Version,OS type,Vendor
<Software ID>,<Software name>,<Version>,<OS type>,<Vendor>
...
```

1.2 Definition of Installed Software Information

Register the installed software information to manage software that is not compatible with UpdateAdvisor (middleware) as installed software or to configure software parameters.

Use swcfmg_installedsoftware (Installed Software Information Management Command) to register installed software information. Perform the operations below (refer to "Chapter 3 Command Reference" for information on the command):

- Registration of installed software information
- Deletion of installed software information
- Output of installed software information

Registration of installed software information

Register the installed software information for the server on which software registered using the Software Information Management Command was installed.

[Registration information]

- IP address <*IP address*>
- Software ID <*Software ID*>

Information about Fujitsu middleware installed in managed servers is registered in the CMDB when Fujitsu middleware discovery is performed (note that no information for software that is not compatible with UpdateAdvisor (middleware) is collected during discovery). The installation status of the software on each server should therefore be registered using the Installed Software Information Management Command.

Information about installed software is registered using the combination of the IP address of the managed server and the software ID.

Specify CSV files in the format below using the Installed Software Information Management Command (refer to "Chapter 4 File Reference" for information on file format):

```
[#]<IP address>,<Software ID>
...
```

Check the software ID using the software information output function.

An error will occur if Systemwalker Software Configuration Manager does not manage the IP address or if the software ID is not registered in it.

Note that an error will also occur if the specified combination of IP address and software ID already exists.

Deletion of installed software information

Delete the installed software information that was registered using the Installed Software Information Management Command.

To delete software information, specify CSV files in the format above using the Software Information Management Command.

An error will occur if Systemwalker Software Configuration Manager does not manage the IP address or if the software ID is not registered in it.

Note that an error will also occur if the specified combination of IP address and software ID contains an IP address that already exists if the software ID has not been configured as installed software.

This command cannot be used to delete information about installed software that was registered during the discovery process.

Output of installed software information

Output the installed software information in the CSV format below (the output destination is the standard output of the command or the specified file):

```
#IP address,Software ID
<IP address>,<Software ID>
...
```

Specify the -all option to output information registered in the discovery process as well as installed software information that was registered using this command. Refer to "Chapter 3 Command Reference" for information on options when executing commands.

Chapter 2 Definition of Parameter Information

The following definitions are required when managing software parameters. Note that these definitions are not required if using the software definitions pre-registered in this product.

- Definition of Parameters to be Set
- Definition of Parameters to be Collected

The settings below are then configured based on these definitions.

- Parameter Value Settings
- Refer to the *Operation Guide* for information on how to set parameter values.

See

There are three types of software definitions:

a. Definitions pre-registered in this product

b. Definitions to be published on the web

c. User-created definitions

Refer to "Middleware List" in the *Parameter Reference* for information on a.

Refer to the software technical information website for information on b.

```
URL:http://software.fujitsu.com/jp/technical/
```

2.1 Definition of Parameters to be Set

Create a parameter settings definition for defining parameters that can be configured in the software. Use the parameter settings definition management command (swcfmg_param_settingdef) to create a parameter settings definition. To associate the software and parameter settings definitions, use the command to associate software and parameter definitions (swcfmg_param_defassoc). Refer to "Chapter 3 Command Reference" for information on the commands.

The knowledge required to create this parameter settings definition is explained below:

- Parameter settings definition

  Specify the list of parameters (key name, type, and default value), method, and script package for this parameter settings definition.

- Parameter types

  The parameter type can be specified as boolean, number, string, string array, or map.

- Variables that can be specified as values

  Variables can also be specified as the default value.

- Parameter settings scripts

  Parameters are configured in the software using a parameter settings script. This script is forwarded to the server and runs on it.

- Package files
- The parameter settings script is converted to a package file compressed in ZIP format and then associated with the parameter settings definition.
- Association with the software

  Although the parameter settings definition defines the parameters that can be configured in the software, it does not specify the corresponding software. For this reason, the parameter settings definition must be associated with the software information.

2.1.1 Parameter Settings Definition

The parameter settings definition is used to define the parameters that can be configured in the software. Specify the list of parameters (key name, type, and default value), method, and script package for this definition.

In the parameter list, specify multiple parameters that can be configured in the software. Each parameter is represented by a key name, type, and default value. However, the default value can also be omitted.

For the parameter setup method, specify which method should be used to configure the parameter values in the software. Select either a batch file parameter settings script that can be used in Windows, or a shell script parameter settings script that can be used in Linux as the method.

A script package is a package file containing the parameter settings script compressed in ZIP format.

Required information

The information to be specified in the parameter settings definition is shown below.

Tag name	Allowable range	Description	Mandatory	Settings
name	256 characters or less	Specifies the parameter settings definition name.	Y	
description	256 characters or less	Specifies the parameter settings definition description.	Y	
method	Select an option	Specifies the parameter setup method script.	Y	Select from the following options: - "cmd": Calls the startup script (startup.cmd). - "sh": Calls the startup script (startup.sh).
parameters	1 or more	Specifies multiple parameters that can be configured in the software.	Y	
key	1 to 256 bytes	Specifies the parameter key.	Y	Characters that can be used are alphanumeric characters, ".", "_", and "-". However, the first character must only be alphabetic.
type	Select an option	Specifies the value type.	Y	Select from the following options: - "boolean": true, false - "number": Number - "string": Character string - "string array": Character string array - "map": (Map) Refer to "2.1.2 Parameter Types" for details.
value	4096 characters or less	Specifies the default value if a value must be set. The value can be changed when	N	Values that can be specified are determined by 'type'.

Tag name		Allowable range	Description	Mandatory	Settings
			configuring this parameter, but it cannot be left blank.		The string "__EMPTY__" (prefixed and suffixed by 2 underscores) cannot be specified. Variables can be specified as values by prefixing them with # (to specify # or \ as part of the value, prefix them with the \ escape character). Refer to "2.1.3 Variables that can be Specified as Values" for details.
	label	64 characters or less	Specifies the label used to display the parameter in the window.	N	
	description	256 characters or less	Specifies the parameter description.	N	
Script package		2 MB or less	Specifies the package file containing the parameter settings script compressed in ZIP format.	Y	For details on parameter settings scripts, refer to "2.1.4 Parameter Settings Scripts". For details on package files, refer to "2.1.5 Package Files".

2.1.2 Parameter Types

A type can be specified for parameters in a parameter settings definition or a parameter collection definition. The following types can be specified:

- boolean

  The values true and false can be specified.

- number

  Numeric values between -2,147,483,648 and 2,147,483,647 can be specified.

- string

  Strings (including empty strings) can be specified.

- string array

  String arrays with zero or more elements can be specified (the array index starts from 1).

- map

  Zero or more pairs (entries) of subkeys and values can be specified as values. Subkeys can contain alphanumeric characters, periods (.), underscores (_), and hyphens (-).

  Example: Specifying values for three subkeys:

```
Subkey: subkey1, value: data1
Subkey: subkey2, value: data2
Subkey: subkey3, value: data3
```

2.1.3 Variables that can be Specified as Values

Variables can be specified as parameter values when specifying a server or OS value (computer name, host name, or IP address) as a parameter value.

Specify variables using the following format:

```
#{<variable name>}
```

Example:

```
<value>#{server.os.computername}</value>
```

List of variables

The variables that can be used are shown below, together with their descriptions:

Variable name	Description
server.os.computername	Computer name/host name
server.nic.ipaddress	IP address of NIC

Note

To specify # or \ as part of the value, prefix them with the \ escape character.

2.1.4 Parameter Settings Scripts

A parameter settings script is a script used to configure parameters in software. This script is forwarded to the server and runs on it.

A parameter settings script is composed of multiple files. Note that the names of Windows batch files and Linux shell scripts are fixed. These files are described below.

- Startup script

  The startup script is always the first script to be called.

- The file name of this script is fixed. It is named startup.cmd (if a batch file) or startup.sh (if a shell script). This script must call the environment variable settings script, which is the input, and return the result, which is the output. Other processes can be created for different software products.

- Environment variable settings script

  Environment variables are used to input parameter values in the startup script. These environment variables are configured using the environment variable settings script.

  This script is named setenv.cmd (if a batch file) or setenv.sh (if a shell script). This product generates the environment variable settings script from the parameters settings definition and parameter information.

- Parameter information XML file for configured parameters

  Input to the startup script is also passed via the parameter information XML file.

  This file is named parameterinfo.xml. This product generates the parameter information XML file from the parameters settings definition and parameter information.

- File attachments
  Any file can be used from the startup script. The file will be forwarded to the server with the startup script.

  If a file attachment includes a shell script, configure execution privileges within the startup script of the shell script.

Startup script format (batch files) [Windows]

Create the startup script for batch files (startup.cmd) using the format below.

The environment variable settings script (setenv.cmd) is called first. Specify a process that acquires parameter values from the environment variables and sets the parameters in the software. Parameter values can also be acquired from the parameter information XML file. The script returns 0 if successful, or another value otherwise. The standard output and standard error output are directed to the agent log.

```
@echo off
setlocal
@rem Configure environment variables
call .\setenv.cmd
@rem Software setup process
```

```
<Processing for each software product>
setup -host %manager_name%
@rem Return results (normal)
if ERRORLEVEL 1 goto ERROR_END
endlocal
exit /B 0
@rem Return results (error)
: ERROR_END
echo ERROR0001 Parameter settings failed. 1>&2
endlocal
exit /B 1
```

Startup script format (shell scripts) [Linux]

Create the startup script for shell scripts (startup.sh) using the format below.

The environment variable settings script (setenv.sh) is called first. Specify a process that acquires parameter values from the environment variables and sets the parameters in the software. Parameter values can also be acquired from the parameter information XML file. The script returns 0 if successful, or another value otherwise. The standard output and standard error output are directed to the agent log.

```
#!/bin/sh
# Configure environment variables
source ./setenv.sh
# Software setup process
<Processing for each software product>
setup -host ${manager_name}
# Return results
if [ $? = "0" ]; then
 # Returns normal
 exit 0
else
 # Returns an error
 echo "ERROR0001 Parameter settings failed." 1>&2
 exit 1
fi
```

Environment variable settings script format (batch files) [Windows]

This product generates the environment variable settings script batch file (setenv.cmd) in the format below.

The environment variable names are determined by the parameter keys in the parameter settings definition (parameter information). The environment variable values are determined by the parameter values in the parameter settings definition or in the parameter information. However, the following conversions are performed:

- Periods (.) and hyphens (-) in keys are converted to underscores (_)

  Environment variable names are converted to strings in which periods (.) and hyphens (-) in keys are converted to underscores (_).

- String array-type environment variables combine the key index with the array index

  An environment variable is created for each element of the array, with the name being the key, followed by underscore, followed by the index. The value of the environment variable name containing the key is the number of elements in the array. If the array has no elements, an environment variable name will be created for the key only. The index starts from 1.

- Map-type environment variables combine the key and subkey

  An environment variable is created for each subkey, with the name being the key, followed by underscore, followed by the subkey. The value of the environment variable name containing the key is the number of entries in the map. If the map has no entries, an environment variable name will be created for the key only.

- Specify __EMPTY__ for empty values

```
set <parm key1>=<parm val>
set <parm key array>=<num of elements>     ...string array type
set <parm key array>_<index> =<parm val>  ...string array type
set <parm key map>=<num of entries>        ...map type
```

```
set <parm key map>_<subkey>=<parm val>    ...map type
<Define as many environment variables as there are parameters>
```

Example:

```
set manager_name=server1
set manager_adata=2                  ...string array type
set manager_adata_1=data1           ...string array type
set manager_adata_2=data2           ...string array type
set manager_bdata=0                  ...string array type with 0 elements
set manager_cdata=3                  ...map type
set manager_cdata_subkey1=data1    ...map type
set manager_cdata_subkey2=data2    ...map type
set manager_cdata_subkey3=data3    ...map type
set manager_ddata=0                  ...map type with 0 entries
```

Environment variable script format (shell scripts) [Linux]

This product generates the environment variable settings script shell script (setenv.sh) in the format below.

The environment variable names are determined by the parameter keys in the parameter settings definition (parameter information). The environment variable values are determined by the parameter values in the parameter settings definition or in the parameter information. However, the following conversions are performed:

- Periods (.) and hyphens (-) in keys are converted to underscores (_)

  Environment variable names are converted to strings in which periods (.) and hyphens (-) in keys are converted to underscores (_).

- String array-type environment variables combine the key index with the array index

  An environment variable is created for each element of the array, with the name being the key, followed by underscore, followed by the index. The value of the environment variable name containing the key is the number of elements in the array. If the array has no elements, an environment variable name will be created for the key only. The index starts from 1.

- Map-type environment variables combine the key and subkey

  An environment variable is created for each subkey, with the name being the key, followed by underscore, followed by the subkey. The value of the environment variable name containing the key is the number of entries in the map. If the map has no entries, an environment variable name will be created for the key only.

- Enclose values that contain spaces in double quotation marks ("")

```
<parm key>=<parm val>
<parm key>=<num of elements>    ...string array type
<parm key>_<index>=<parm val>   ...string array type
<parm key>=<num of entries>     ...map type
<parm key>_<subkey>=<parm val>  ...map type
<Define as many environmental variables as there are parameters>
```

Example:

```
manager_name=server1
manager_adata=2                   ...string array type
manager_adata_1=data1            ...string array type
manager_adata_2=data2            ...string array type
manager_bdata=0                   ...string array type with 0 elements
manager_cdata=3                   ...string array type
manager_cdata_subkey1=data1    ...map type
manager_cdata_subkey2=data2    ...map type
manager_cdata_subkey3=data3    ...map type
manager_ddata=0                   ...map type with 0 entries
```

Parameter information XML file format

The parameter information XML file for configured parameters is generated by this product. Reference the content of the key tags and their values <parameters> section of this file. Other tags are omitted or are not required for referencing.

- Tag to be referenced

  parameters

- Tag to be omitted

  description

- Tag not required for referencing

  name

An example of a generated parameter information XML file is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterInfo version="3.0">
    <name>parmInfoName</name>
    <parameters>
        <parameter>
            <key>parmKey</key>
            <value>parmVal</value>
        </parameter>
        ...
    </parameters>
</parameterInfo>
```

An example of a parameter information XML file is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterInfo version="3.0">
    <name>Parameter Info</name>
    <parameters>
        <parameter>
            <key>manager.name</key>
            <value>server1</value>
        </parameter>
        <parameter>
            <key>manager.adata</key>
            <array>
                <element>data1</element>
                <element>data2</element>
            </array>
        </parameter>
        <parameter>
            <key>manager.cdata</key>
            <map>
                <entry>
                    <subkey>subkey1</subkey>
                    <value>data1</value>
                </entry>
                <entry>
                    <subkey>subkey2</subkey>
                    <value>data2</value>
                </entry>
                <entry>
                    <subkey>subkey3</subkey>
                    <value>data3</value>
                </entry>
            </map>
        </parameter>
    </parameters>
</parameterInfo>
```

Note

- **Commands that cannot be used with a script**

  Do not execute the following commands from a script, because this will cause the script to enter standby status on the business server, and its processing will not complete:

  - Commands that require interaction [Windows/Linux]
  - Commands for which a window opens during execution [Windows]
  - AT commands [Windows]
  - Shell scripts created using PowerShell [Windows]
  - Commands running in full-screen mode [Linux]

- **Parameter settings scripts**

  - Order of execution

    The parameter settings script is operated according to the order specified in the Parameter Settings Wizard.

- **Startup script**

  - Execution privileges

    Must be executed by an Administrator in Windows.

    Must be executed by a superuser in Linux.

  - Current directory

    The current directory is the path in which the startup script files are stored.

  - Execution privileges [Linux]

    Configured automatically during execution.

  - Line feed

    In Windows, use CR+LF.

    In Linux, use LF.

  - Byte order mark (BOM) [Linux]

    Do not include the UTF-8 byte order mark (BOM) in shell scripts.

- **Environment variable settings script**

  - Environment variable definitions

    Environment variables cannot be defined if parameter values are not configured. Accordingly, whether parameter values will be configured in the startup script is determined by whether environment variables have been defined.

    The method for determining whether environment variables have been defined is shown below:

    Batch files: [Windows]

    ```
    if defined <environment variable> (<defined process>) else <undefined process>
    ```

    Example:

    ```
    set PARAM=%hostname%
    if defined parameter_Key1 (set PARAM=-v %parameter_Key1% %PARAM%)
    ```

    Shell scripts: [Linux]

    ```
    ${<environment variable>+${<environment variable>}}
    ```

    Example:

```
PARAM=${hostname}
PARAM="${parameter_Key1+"-v ${parameter_Key1}"} ${PARAM}"
```

- Empty strings in values [Windows]

  An empty string cannot be configured in environment variables in Windows. For this reason, specify empty strings in batch files using "__EMPTY__" (prefixed and suffixed by 2 underscores). Note that as a result, the string "__EMPTY__" cannot be used as a parameter value.

- Symbols (special characters)

  Some parameter values passed by an environment variable settings script do not accept symbols (special characters), or an escape character must be specified before a symbol (special character).

- **File attachments**

  - Execution privileges [Linux]

    If a file attachment includes a shell script, configure execution privileges within the startup script of the shell script.

2.1.5 Package Files

A package file refers to one or more compressed files in ZIP format. These files are used to associate the parameter settings script with the parameter settings definition and parameter information. Package files are decompressed on the server once they have been forwarded.

The package file to be associated with the parameter settings definition is known as the script package. The package file registered in the parameter information is known as the parameter package. The parameter settings scripts contained in each of these package files is different. An explanation is provided below:

- Script package

  Includes the startup script of the parameter settings script, and the file attachment. The startup script must be included.

- Parameter package

  Includes the file attachment of the parameter settings script.

Script package

A script package is a ZIP file that contains a startup script and a file attachment. This ZIP file must be stored using the structure shown below. Always store the startup script in the ZIP file root. The file attachment can contain a directory.

```
<ZIP file root>
+ <Startup script (startup.cmd or startup.sh)>
+ <File attachment>
```

Parameter package

A parameter package is a ZIP file that contains a file attachment. This ZIP file must be stored using the structure shown below. The file attachment can contain a directory.

```
<ZIP file root>
+ <File attachment>
```

Execution example of the parameter settings script

Script packages and parameter packages are forwarded to the server and decompressed on it. The environment variable settings script generated by this product is also forwarded and stored in the work directory on the server as described below - note that the directory in which the startup script is stored becomes the current directory, and the startup script is executed.

```
<work directory>
+ <software id>
  + scriptpkg
  |  + Startup script (startup.cmd or startup.sh)
  |  + Environment variable settings script (setenv.cmd, setenv.sh)
```

```
 | + File attachment in the script package
 + paramdata
 | + Parameter information XML file (parameterinfo.xml) for configured parameters
 + parampkg
     + Parameter package attachment
```

A description of each directory is shown below:

- Software ID

  Directory unique to each software product, with the name *<software id>*.

- scriptpkg

  Stores the startup script, the file attachment (both included in the script package) and the environment variable settings script.

- paramdata

  Stores the parameter information XML file (parameterinfo.xml) for configured parameters.

- parampkg

  Stores the file attachment included in the parameter package. When referring to this file in the startup script, use the relative path.

- **Package files**
  - Store the startup script in the ZIP file root, not in the directory, otherwise it cannot be executed.
  - Package files forwarded to the server are deleted once the script has been executed.

2.1.6 Association with the Software

Although the parameter settings definition defines the parameters that can be configured in the software, it does not specify the corresponding software. For this reason, the parameter settings definition must be associated with the software information.

Multiple software information items can be associated for each parameter settings definition. For this reason, parameter settings definitions can be aggregated as a single item, even if there are multiple editions of the software. However, since Windows and Linux editions cannot coexist, distinct parameter settings definitions must be created in such cases. Note that it is also possible to associate multiple parameter settings definitions with software information.

2.2 Definition of Parameters to be Collected

Create a parameter collection definition for defining parameters that can be collected from the software. Use the parameter collection definition management command (swcfmg_param_collectingdef) to create a parameter collection definition. To associate the software and parameter collection definitions, use the command to associate software and parameter definitions (swcfmg_param_defassoc). Refer to "Chapter 3 Command Reference" for information on the commands.

The knowledge required to create this parameter collection definition is explained below.

- Parameter collection definition

  Use the parameter collection definition to define parameters that can be collected from the software. Specify the list of parameters (key name and type), method, and script package for this parameter collection definition.

- Parameter types

  The parameter type can be specified as boolean, number, string, string array, or map (refer to "2.1.2 Parameter Types" for details).

- Parameter collection scripts

  Parameters are collected from the software using a parameter collection script. This script is forwarded to the server and runs on it. Information about the parameters collected from the software is output to files such as the parameter information XML file (parameterinfo.xml). This file is transferred to an admin server.

- Package files

  The parameter collection script is converted to a package file compressed in ZIP format and then associated with the parameter collection definition.

- Association with the parameter collection definition

  Although the parameter collection definition defines the parameters that can be collected from the software, it does not specify the corresponding software. For this reason, the parameter collection definition must be associated with the software information.

  If multiple parameter collection definitions are to be associated with a single software information item, you cannot specify a duplicate key for the parameter collection definitions.

2.2.1 Parameter Collection Definition

The parameter collection definition is used to define the parameters that can be collected from the software. Specify the list of parameters (key name and type), method, and script package for this definition.

In the parameter list, specify multiple parameters that can be collected from the software. Each parameter is represented by a key name and type.

For the parameter collection method, specify which method should be used to collect the parameter values from the software. Select either a Windows batch file or Linux shell script.

A script package is a package file containing the parameter collection script compressed in ZIP format.

Specified details

An explanation of details to be specified in the parameter collection definition is shown below.

Tag name	Allowable range	Description	Mandatory	Settings
name	256 characters or less	Specifies the parameter collection name.	Y	
method	Selection	Specifies the parameter collection method.	Y	Select from the following options: - "cmd": Calls the 'discover.cmd' discovery script. - "sh": Calls the 'discover.sh' discovery script.
parameters	1 or more	Specifies multiple parameters that can be collected from the software.	Y	
key	1 to 256 bytes	Specifies the parameter key.	Y	Characters that can be used are alphanumeric characters, ".", "_", and "-". However, the first character must only be alphabetic.
type	Select an option	Specifies the value type.	Y	Select one of the following options: - "boolean" - "number" - "string" An empty string can also be specified.

Tag name		Allowable range	Description	Mandatory	Settings
					- "string array" Empty arrays can also be specified. - "map" Empty maps can also be specified. Refer to "2.1.2 Parameter Types" for details.
	label	64 characters of less	Specifies the label used to display the parameter in the window.	N	
Script package		2 MB or less	Specifies the package file containing the parameter collection script compressed in ZIP format.	Y	For details on parameter collection scripts, refer to "2.2.2 Parameter Collection Scripts". For details on package files, refer to "2.1.5 Package Files".

2.2.2 Parameter Collection Scripts

A parameter collection script is a script used to collect parameters from the software. This script is forwarded to the server and runs on it. Information about the parameters collected from the software is output to files such as the parameter information XML file (parameterinfo.xml). This file is transferred to an admin server.

A parameter collection script is composed of multiple files. Note that the names of Windows batch files and Linux shell scripts are fixed. These files are described below.

- Discovery script

  The discovery script is always the first script to be called.

  This script is named discover.cmd (if a batch file) or discover.sh (if a shell script). This script must output information about the parameters collected from the software to the parameter information XML file (parameterinfo.xml). Other processes can be created for each different software product.

- File attachments
  Any file can be used from the discovery script. The file and the discovery script will be forwarded to the server.

  If a file attachment includes a shell script, configure execution privileges within the discovery script of the shell script.

The discovery script outputs information about parameters collected from the software to a file. A description of this file is shown below.

- Collected parameter information XML file

  XML file containing information about collected parameters. This file has the fixed name parameterinfo.xml.

- File attachment for collected parameters

  Any file can be used for parameter information. This file is handled as an attachment.

Discovery script format (batch files) [Windows]

Create the discovery script for batch files (discover.cmd) using the format below.

The -dir option must specified as the first argument (%1), and the directory to which the parameter information should be output must be specified as the second argument (%2). Information about the parameters collected in this directory is output to the parameter information XML file (parameterinfo.xml) and to the file attachment. The parameter information XML file must be output. However, depending on the specifications of the parameter collection definition, the file attachment might not always be output correctly. The script returns 0 if successful, or another value otherwise. The standard output and standard error output are directed to the agent log.

```
@echo off
setlocal
@rem Process to collect parameters from software
<Processing for each software product>
<Process to output parameter information to directory specified as %2 >
@rem Return results (normal)
if ERRORLEVEL 1 goto ERROR_END
endlocal
exit /B 0
@rem Return results (error)
: ERROR_END
echo ERROR0002 Parameter collecting failed. 1>&2
endlocal
exit /B 1
```

Discovery script format (shell scripts) [Linux]

Create the discovery script for shell scripts (discover.sh) using the format shown below.

The -dir option must be specified as the first argument ($1), and the directory to which the parameter information should be output must be specified as the second argument ($2). Information about the parameters collected in this directory is output to the parameter information XML file (parameterinfo.xml) and to the file attachment. Output is not required when there is no file attachment. The script returns 0 if successful, or another value otherwise. The standard output and standard error output are directed to the agent log.

```
#!/bin/sh
# Process to collect parameters from software
<Processing for each software product>
<Process to output parameter information to directory specified as $2 >
# Return results
if [ $? = "0" ]; then
 # Returns normal
 exit 0
else
 # Returns an error
 echo "ERROR0002 Parameter collecting failed." 1>&2
 exit 1
fi
```

Output directory for discovery script

When using the discovery script to output parameter information, output using the directory format shown below. A parameter information XML file must be created. Only create a file attachment if required.

```
+ <Output directory>
  + Collected parameter information XML file (parameterinfo.xml)
  + File attachment for collected parameters
```

Parameter information XML file format

Create the discovery script so that the parameters collected by this script are output to the parameter information XML file. In this parameter information XML file, specify values for the tags of the keys and values in the parameters. For other tags, specify any value, or omit altogether.

- Tag for which value is specified

  parameters

- Tag to be omitted

  description

- Tag for which any value is specified

  name

An example of the parameter information XML file output is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterInfo version="3.0">
    <name><parm info name></name>
    <parameters>
        <parameter>
            <key><parm key></key>
            <value><parm val></value>
        </parameter>
        ...
    </parameters>

</parameterInfo>
```

For examples of parameter information XML files, refer to "2.1.4 Parameter Settings Scripts".

Note

- **Commands that cannot be used with a script**

  Do not execute the following commands from a script, because this will cause the script to enter standby status on the business server, and its processing will not complete:

  - Commands that require interaction [Windows/Linux]
  - Commands for which a window opens during execution [Windows]
  - AT commands [Windows]
  - Shell scripts created using PowerShell [Windows]
  - Commands running in full-screen mode [Linux]

- **Parameter collection scripts**

  - Order of execution

    The parameter collection scripts do not operate in any particular order.

- **Discovery scripts**

  - Execution privileges

    Must be executed by an Administrator in Windows.

    Must be executed by a superuser in Linux.

  - Current directory

    The current directory is the path in which the discovery script files are stored.

  - Execution privileges [Linux]

    Configured automatically during execution.

  - Line feed

    In Windows, use CR+LF.

    In Linux, use LF.

  - Byte order mark (BOM) [Linux]

    Do not include the UTF-8 byte order mark (BOM) in shell scripts.

- **File attachments**

  - Execution privileges [Linux]

    If a file attachment includes a shell script, configure execution privileges within the discovery script of the shell script.

2.2.3 Package Files

A package file refers to one or more compressed files in ZIP format. These files are used to associate the parameter collection script with the parameter collection definition. Package files are decompressed on the server once they have been forwarded. They are also used to store file attachments, which are output by the discovery script of the parameter collection script, on the admin server.

The package file to be associated with the parameter collection definition is known as the script package. The compressed file in ZIP format containing the file attachment output by the discovery script of the parameter collection script is known as the parameter package. The parameter collection scripts contained in each of these package files is different. An explanation is provided below:

- Script package

  Includes the discovery script of the parameter collection script, and the file attachment. The discovery script must be included.

- Parameter package

  Includes the file attachment output by the discovery script of the parameter collection script. The parameter information XML file (parameterinfo.xml) is not included. Note that no parameter package will be created if there is no file attachment.

Script package

A script package is a ZIP file that contains a discovery script and a file attachment. This ZIP file must be stored using the structure shown below. Always store the discovery script in the ZIP file root. The file attachment can contain a directory.

```
<ZIP file root>
+ <Discovery script (discover.cmd or discover.sh)>
+ <File attachment>
```

Parameter package

A parameter package is a ZIP file that contains a file attachment. This ZIP file must be stored using the structure shown below. It also contains a directory if the file attachment contains a directory.

```
<ZIP file root>
+ <File attachment>
```

Execution example of the parameter collection script

The script package is forwarded to the server and decompressed on it. The parameter collection script is stored in the work directory on the server as described below - note that the directory in which the discovery script is stored becomes the current directory, and the discovery script is executed.

```
<work directory>
+ <software id>
 + Discovery script (discover.cmd or discover.sh)
 + Script package file attachment
```

A description of each directory is provided below:

- Software ID directory

  Directory unique to each software product, with the name *<software id>*.

  Stores the discovery script and the file attachment included in the script package.

The discovery script outputs the parameter information to the directory below (the file attachment for the collected parameters is included in the parameter package).

```
+ <output directory>
  + Collected parameter information XML file (parameterinfo.xml)
  + File attachment for collected parameters
```

Note

- **Package files**
  - Store the discovery script in the ZIP file root, not in the directory, otherwise it cannot be executed.
  - Package files forwarded to the server are deleted once the script has been executed.

2.2.4 Association with the Software

Although the parameter collection definition defines the parameters that can be collected from the software, it does not specify the corresponding software. For this reason, the parameter collection definition must be associated with the software information.

Multiple software information items can be associated for each parameter collection definition. For this reason, parameter collection definitions can be aggregated as a single item, even if there are multiple editions of the software. However, since Windows and Linux editions cannot coexist, distinct parameter collection definitions must be created in such cases.

Moreover, multiple parameter collection definitions can be associated with a single software information item. However, you cannot specify a duplicate key in the parameter list for these parameter collection definitions. Even if you associate multiple parameter collection definitions, one parameter list is collected for one software product.

Chapter 3 Command Reference

3.1 Command List

The following table lists commands.

Type	Function	Command	Overview
Software management commands	Software information management command	swcfmg_software	Registers, updates, deletes, and lists software information managed by Systemwalker Software Configuration Manager.
	Installed software information management command	swcfmg_installedsoftware	Registers, deletes, and lists software information installed on the managed server.
Parameter management commands	Parameter settings definition management command	swcfmg_param_settingdef	Adds, updates, deletes, lists, and acquires parameter settings definitions. Also, uploads or downloads the script package associated with parameter settings definitions.
	Parameter collection definition management command	swcfmg_param_collectingdef	Adds, updates, deletes, lists, and acquires parameter collection definitions. Also, uploads or downloads the script package associated with the script package of a parameter collection definition.
	Command to associate software and parameter definitions	swcfmg_param_defassoc	Associates the software and parameter settings definitions, or the software and parameter collection definition. The command can also be used to cancel such associations, as well as display a list of associations.

Note

- Some commands provided by Systemwalker Software Configuration Manager cannot be executed simultaneously.

  An exclusive control error will occur if a command is executed while another command is still executing.

- In order to execute the commands provided by Systemwalker Software Configuration Manager, the environment variables that are set up during installation must be enabled.

  For this reason, execute commands from a command prompt that is opened after Systemwalker Software Configuration Manager has been installed.

3.2 Software Management Commands

This section explains the software management commands.

3.2.1 swcfmg_software (Software Information Management Command)

Description

Registers, updates, deletes, and lists software information managed by Systemwalker Software Configuration Manager.

Use this command to register software information to ensure that software not collected by discovery (software incompatible with UpdateAdvisor (middleware)) will be displayed as installed software on the management console, or to allow software parameters to be configured.

Synopsis

[Windows]

```
swcfmg_software.exe
{ -add -file <Software information file path> [-e <Character encoding>] |
-update -file <Software information file path> [-e <Character encoding>] |
-delete -id <Software ID>,... |
-list [ -file <Output file path>] [-e <Character encoding>] }
```

[Linux]

```
swcfmg_software
{ -add -file <Software information file path> [-e <Character encoding>] |
-update -file <Software information file path> [-e <Character encoding>] |
-delete -id <Software ID>,... |
-list [-file <Output file path>] [-e <Character encoding>] }
```

Options

-add -file <*Software information file path*>

Registers software information using the specified software information file

The software information file is in CSV format. Refer to "4.2.1 Software Information File" for information on software information files.

-update -file <*Software information file path*>

Updates software information using the specified software information file

The software information file is in CSV format. Refer to "4.2.1 Software Information File" for information on software information files.

-delete -id <*Software ID*>,...

Deletes registered software information. Multiple software IDs can be specified by delimiting them with ',' (comma).

The software ID is automatically set during software information registration. Specifying -list allows the software ID to be checked from the software information output.

-list [-file <*Output file path*>]

Outputs registered software information in CSV format.

If -file <*Output file path*> is omitted, information will be output to the standard output for the command.

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If the file cannot be output because of an error, either a new file will not be created or, if an existing file was to be overwritten, the existing file will be kept as it is, without being overwritten.

-e <*Character encoding*>

Specify the character encoding for the file.

The character encodings below can be specified (single-byte uppercase characters must be used):

- UTF-8

If this option is omitted, the character encoding of the execution environment of this command will be used.

Return value

Return value	Meaning
0	Completed normally.
10	A parameter error has occurred.
20	There is an error in the specified I/O file path.
30	The specified character encoding is incorrect.
40	Failed to read the software information file.
50	The software information file description is incorrect.
70	The specified software ID does not exist.
80	The software information targeted for deletion cannot be deleted because the parameter settings information or the installed software information has been registered.
100	Systemwalker Software Configuration Manager is not running.
101	A command that cannot be executed at the same time as this command is executing.
102	You do not have the privileges required to execute this command.
110	A write error has occurred.
200	Setup has not been performed.
255	A system error has occurred.

Command location

Admin server

[Windows]

<*Systemwalker Software Configuration Manager installation directory*>\SWCFMGM\bin

[Linux]

/opt/FJSVcfmgm/bin

Privilege required for execution

[Windows]

- Administrator privileges are required. If using Windows Server 2008 or later as your operating system, execute this command as an administrator.
- This command can be executed on the admin server.

[Linux]

- This command can only be executed by the system administrator (superuser).
- This command can be executed on the admin server.

Notes

Common notes

- This command cannot be executed simultaneously with other commands. An exclusive control error will occur if a command is executed while another command is still executing.
- Execute this command while Systemwalker Software Configuration Manager is running.

Notes regarding registration

- Software information is registered in accordance with the software information file.

  An error will occur in the input file if the description includes <*Software ID*>. Do not include <*Software ID*> when registering software information for the first time.

- An error will occur if information that has the same <Product name>, <Version> and <Platform> is judged to be all the same software and is already registered.
- Behavior is not guaranteed if the character encoding specified for this command does not match the character encoding used in the software information file. Ensure that the character encoding matches.

Notes regarding updating

- When updating registered software information, it is advisable to edit the file that is listed, then treat the edited file as the input file for updating.
- An error will occur in the input file if the description does not include <*Software ID*>. When updating, always include <*Software ID*>.

Notes regarding deletion

- Use this command to delete software information that no longer needs to be managed by Systemwalker Software Configuration Manager.
- Before deleting software information, ensure that there is no installation information for the software in question, nor any parameter settings information or parameter collection information. The software information cannot be deleted if such information exists.

Notes regarding list display

- If no software information has been registered, only the header information will be output.

```
#Software ID,Software name,Version,OS type,Vendor
```

Notes regarding registration and update errors

- If there is an error in the software information file, registration and update of the software information will be canceled. An error message similar to the following will be output to the standard output:

```
The file description is incorrect.[Line number:<n>](Cause:<Cause>)
```

  One of the following messages will be output in <*Cause*>:

Message	Meaning
Number of elements.	Review the number of elements in the line indicated.
Mandatory elements are not specified.	Enter the mandatory items in the line indicated.
The same software is specified in another line.	The same software information is already specified in one of the preceding lines.
The specified elements are invalid. (Software ID)	The (Software ID) in the line indicated is invalid. The (Software ID) cannot be set at registration.
The specified software does not exist.	Software information corresponding to the (Software ID) in the line indicated does not exist. Specify the existing software ID.
The software has already been registered.	The software information in the line indicated has already been registered.

Examples

- Registration

  [Windows]

  ```
  swcfmg_software.exe -add -file C:\work\softwareinfo.csv
  ```

  [Linux]

  ```
  swcfmg_software -add -file /tmp/softwareinfo.csv
  ```

- Update

  [Windows]

  ```
  swcfmg_software.exe -update -file C:\work\softwareinfo.csv
  ```

  [Linux]

  ```
  swcfmg_software -update -file /tmp/softwareinfo.csv
  ```

- Deletion

  [Windows]

  ```
  swcfmg_software.exe -delete -id UDP00001,UDP00002,UDP00003
  ```

  [Linux]

  ```
  swcfmg_software -delete -id UDP00001,UDP00002,UDP00003
  ```

- List display

  [Windows]

  ```
  swcfmg_software.exe -list -file C:\work\softwareinfo.csv
  ```

  [Linux]

  ```
  swcfmg_software -list -file /tmp/softwareinfo.csv
  ```

Execution results/output format

- List display

  Standard output

  ```
  #Software ID,Software name,Version,OS type,Vendor
  UDP000001,ABCDE-Software,1.0,windows,ABCDE Company
  ```

  File output

  ```
  Output of the software information has started.
  Output of the software information has ended.
  ```

  Note that the following header information is added to the first line of the CSV file that is output:

  ```
  #Software ID,Software name,Version,OS type,Vendor
  ```

3.2.2 swcfmg_installedsoftware (Installed Software Information Management Command)

Description

Registers, deletes, and lists software information installed on the managed server.

Use this command to register installed software information to ensure that software not collected by discovery (software incompatible with UpdateAdvisor (middleware)) will be displayed as installed software on the management console, or to allow software parameters to be configured.

Synopsis

[Windows]

swcfmg_installedsoftware.exe

{ -add -file <*Installed software information file path*> [-e <*Character encoding*>] |

-delete -file <*Installed software information file path*> [-e <*Character encoding*>] |

-list [-ip <*IP address*>,...] [-file <*Output file path*>] [-e<*Character encoding*>] [-all] }

[Linux]

swcfmg_installedsoftware

{ -add -file <*Installed software information file path*> [-e <*Character encoding*>] |

-delete -file <*Installed software information file path*> [-e <*Character encoding*>] |

-list [-ip <*IP address*>,...] [-file <*Output file path*>] [-e<*Character encoding*>] [-all] }

Options

-add -file <*Installed software information file path*>

Registers installed software information using the specified installed software information file

The installed software information file is in CSV format. Refer to "4.2.2 Installed Software Information File" for information on installed software information files.

-delete -file <*Installed software information file path*>

Deletes installed software information using the specified installed software information file

The installed software information file is in CSV format. Refer to "4.2.2 Installed Software Information File" for information on installed software information files.

-list [-ip <*IP address*>,...] [-file <*Output file path*>] [-all]

Use this command to output registered installed software information in CSV format.

[-ip <*IP address*>,...]

Specifies the IP address of the server that outputs the installed software information.

If this option is omitted, the installed software information on all servers will be output.

[-file <*Output file path*>]

Specify the output file path.

If this option is omitted, information will be output to the standard output for the command.

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If output to the file fails due to an error, no new file will be created. If an existing file was to be overwritten, it will be kept as is without being overwritten.

[-all]

Use this command to output not only registered information, but also all installed software information for each server, including discovered information.

-e <*Character encoding*>

Specify the character encoding for the file.

The character encodings below can be specified (single-byte uppercase characters must be used):

- UTF-8

If this option is omitted, the character encoding of the execution environment of this command will be used.

Return value

Return value	Meaning
0	Completed normally.
10	A parameter error has occurred.
20	There is an error in the specified I/O file path.
30	The specified character encoding is incorrect.
40	Failed to read the installed software information file.
50	The installed software information file description is incorrect.
100	Systemwalker Software Configuration Manager is not running.
101	A command that cannot be executed at the same time as this command is executing.
102	You do not have the privileges required to execute this command.
110	A write error has occurred.
200	Setup has not been performed.
255	A system error has occurred.

Command location

Admin server

[Windows]

<*Systemwalker Software Configuration Manager installation directory*>\SWCFMGM\bin

[Linux]

/opt/FJSVcfmgm/bin

Privilege required for execution

[Windows]

- Administrator privileges are required. If using Windows Server 2008 or later as your operating system, execute this command as an administrator.
- This command can be executed on the admin server.

[Linux]

- This command can only be executed by the system administrator (superuser).
- This command can be executed on the admin server.

Notes

Common notes

- This command cannot be executed simultaneously with other commands. An exclusive control error will occur if a command is executed while another command is still executing.
- Execute this command while Systemwalker Software Configuration Manager is running.

Notes regarding registration

- An error will occur if the server with the IP address listed in the installed software information file is not registered in Systemwalker Software Configuration Manager as a managed server.
- An error will occur if software information with the software ID listed in the installed software information file is not registered in Systemwalker Software Configuration Manager.
- An error will occur if a combination of registered IP address and software ID already exists in the installed software information file.
- Behavior is not guaranteed if the character encoding specified for this command does not match the character encoding used in the software information file. Ensure that the character encoding matches.

Notes regarding deletion

- Use this command to delete installed software information if the installed software information was erroneously set for the managed server or the software was uninstalled from the server in question.
- Use this command to delete the installed software information if a managed server registered in Systemwalker Software Configuration Manager is deleted.
- An error will occur if the server with the IP address listed in the installed software information file is not registered in Systemwalker Software Configuration Manager as a managed server.
- An error will occur if software information with the software ID listed in the installed software information file is not registered in Systemwalker Software Configuration Manager.
- An error will occur if, in the combination of IP address and software ID specified in the installed software information file, the given IP address exists but the software ID has not been set as installed software.
- Behavior is not guaranteed if the character encoding specified for this command does not match the character encoding used in the software information file. Ensure that the character encoding matches.

Notes regarding list display

- If no software information has been registered, only the header information will be output.

```
#Server name,L-platform name,IP-address,Software ID,Software name,Version,OS type,Vendor
```

Notes regarding registration and deletion errors

- If there is an error in the software information file, registration and deletion of the software information will be canceled. An error message similar to the following will be output to the trace log:

```
The information in line n is incorrect.(Cause:<Cause>)
```

One of the following messages will be output in <*Cause*>:

Message	Meaning
Number of elements.	Review the number of elements in the line indicated.
Mandatory elements are not specified.	Enter the mandatory items in the line indicated.
The same IP-address and software id are specified in another line.	A combination of the same IP address and software ID is already specified in one of the preceding lines.
The specified server does not exist.	The server corresponding to the (IP-address) in the line indicated does not exist. Specify the IP address of the existing server.
The specified software does not exist.	Software information corresponding to the (Software ID) in the line indicated does not exist. Specify the existing software ID.

Message	Meaning
The installed software has already been registered.	The software information in the line indicated has already been registered. If you re-register a managed server, the installed software information that was registered previously may still exist. In this case, use the -list option to check the installed software information and delete the relevant installed software information before re-registering it.
The specified installed software does not exist.	The combination of IP address and software ID in the line indicated has not been registered as installed software information.

Examples

- Registration

  [Windows]

  ```
  swcfmg_installedsoftware.exe -add -file C:\work\installedsoftwareinfo.csv
  ```

  [Linux]

  ```
  swcfmg_installedsoftware -add -file /tmp/installedsoftwareinfo.csv
  ```

- Deletion

  [Windows]

  ```
  swcfmg_installedsoftware.exe -delete -file C:\work\installedsoftwareinfo.csv
  ```

  [Linux]

  ```
  swcfmg_installedsoftware -delete -file /tmp/installedsoftwareinfo.csv
  ```

- List display

  [Windows]

  ```
  swcfmg_installedsoftware.exe -list -file C:\work\installedsoftwareinfo.csv
  ```

  [Linux]

  ```
  swcfmg_installedsoftware -list -file /tmp/installedsoftwareinfo.csv
  ```

Execution results/output format

- List display

  Standard output

  ```
  #Server name,L-platform name,IP-address,Software ID,Software name,Version,OS type,Vendor
  Server01,L-Platform01,10.10.10.11,UDP000001,ABCDE-Software,1.0,Windows,ABCDE Company
  Server01,L-Platform01,10.10.10.11,UDP000002,FGHIJ-Software,1.0,Windows,ABCDE Company
  Server02,L-Platform02,10.10.10.12,UDP000001,ABCDE-Software,1.0,Windows,ABCDE Company
  ```

  File output

  ```
  Output of the software information has started.
  Output of the software information has ended.
  ```

  Note that the following header information is added to the first line of the CSV file that is output:

  ```
  #Server name,L-platform name,IP-address,Software ID,Software name,Version,OS type,Vendor
  ```

3.3 Parameter Management Commands

This section explains the parameter management commands.

3.3.1 swcfmg_param_settingdef (Parameter Settings Definition Management Command)

Description

Adds, updates, deletes, lists, and acquires parameter settings definitions. Also, uploads or downloads the script package associated with parameter settings definitions.

Use a parameter settings definition input file as the input file for adding, updating, or acquiring. The parameter settings definition input file is in XML format. Refer to "2.1 Definition of Parameters to be Set" for details.

Uses a script package for uploading and downloading. The script package is a zip file. Refer to "2.1.5 Package Files" for details.

Synopsis

[Windows]

```
swcfmg_param_settingdef.exe
{ -add -file <Parameter settings definition file path> |
-update -id <Parameter settings definition ID> -file <Parameter settings definition file path> |
-delete -id <Parameter settings definition ID>,... |
-list [-file <Output file path>] [-e <Character encoding>] |
-get -id <Parameter settings definition ID> -file <Output file path> |
-upload -id <Parameter settings definition ID> -file <Script package file path> |
-download -id <Parameter settings definition ID> -dir <Output directory path> }
```

[Linux]

```
swcfmg_param_settingdef
{ -add -file <Parameter settings definition file path> |
-update -id <Parameter settings definition ID> -file <Parameter settings definition file path> |
-delete -id <Parameter settings definition ID>,... |
-list [-file <Output file path>] [-e <Character encoding>] |
-get -id <Parameter settings definition ID> -file <Output file path> |
-upload -id <Parameter settings definition ID> -file <Script package file path> |
-download -id <Parameter settings definition ID> -dir <Output directory path> }
```

Options

-add -file *<Parameter settings definition file path>*

Adds the parameter settings definition using the specified parameter settings definition file. The input file is a parameter settings definition XML file.

A parameter settings definition ID is generated for the newly added parameter settings definition. Outputs this ID to standard output.

- Format of ID to be generated

  The parameter settings definition ID is automatically assigned a number according to the following rule:

```
SS (prefix) + 8-digit number
Example:
SS00000001
```

- Output format for standard output

  The parameter settings definition ID is output in the following format:

```
[<Parameter settings definition ID>]
```

-update -id <*Parameter settings definition ID*> -file <*Parameter settings definition file path*>

Updates the parameter settings definition that has the specified ID, using the specified parameter settings definition file path. The input file is a parameter settings definition XML file.

-delete -id <*Parameter settings definition ID*>,

Deletes the parameter settings definition from the parameter settings definition ID. Also, deletes the association (if any) between the software and the parameter settings definition. Multiple parameter settings definition IDs can be specified by delimiting them with ',' (comma).

-list [-file <*Output file path*>]

Uses a parameter settings definition list file in CSV format to output a list of registered parameter settings definitions.

If -file <*Output file path*> is omitted, information will be output to the standard output for the command.

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If the file cannot be output because of an error, either a new file will not be created or, if an existing file was to be overwritten, the existing file will be kept as it is, without being overwritten.

- Output format for standard output

  Outputs the list of parameter settings definitions in the following format:

Item No.	Item	Remarks
1	Parameter settings definition ID	Outputs the parameter settings definition ID.
2	Is the definition predefined in the product?	Outputs whether the parameter settings definition is predefined in the product.
3	Parameter settings definition name	Outputs the parameter settings definition name.
4	Parameter settings method	Outputs the parameter settings method.
5	Script package file name	Outputs the name of the script package file.

- Header format

  Insert the header in the first line of the CSV file using the format shown below. The header is output even if there is no information to be output to the file.

```
#Definition ID,Product predefined,Definition name,Method,Package name
```

-get -id <*Parameter settings definition ID*> -file <*Output file path*>

Outputs the parameter settings definition that has the specified ID to the specified output file path. The output file is a parameter settings definition XML file.

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If output to the file fails due to an error, no new file will be created. If an existing file was to be overwritten, it will be kept as is without being overwritten.

-upload -id <*Parameter settings definition ID*> -file <*Script package file path*>

Uploads the script package to the parameter settings definition that has the specified ID. The input file is a script package zip file.

Script packages that can be uploaded are subject to the following conditions. An error will occur if conditions other than these are used.

- File extension must be .zip.

  Ensure that the file extension is .zip.

- File size must be no greater than 2 MB.

  The maximum uploadable file size is 2 MB.

- Store the startup script in the root.

  Store the startup script (startup.cmd or startup.sh) in the root of the zip file rather than under the directory.

-download -id <*Parameter settings definition ID*> -dir <*Output directory path*>

Downloads the script package registered in the parameter settings definition that has the specified ID. Outputs script package zip files to the output directory path.

Ensure that no files exist in the output directory, or an error will occur.

-e <*Character encoding*>

Specify the character encoding for the file.

The character encodings below can be specified (single-byte uppercase characters must be used):

- UTF-8

If this option is omitted, the character encoding of the execution environment of this command will be used.

Return value

Return value	Meaning
0	Completed normally.
10	A parameter error has occurred.
20	The specified I/O path is incorrect.
30	The specified character encoding is incorrect.
40	Unable to analyze the file.
50	The file description is incorrect.
70	The specified ID does not exist.
71	The specified ID is product-predefined.
72	The package has not been registered in the specified ID.
73	The package has not been correctly registered in the specified ID.
80	Cannot be deleted because there is a predefined parameter associated with the parameter settings definition targeted for deletion.
100	Systemwalker Software Configuration Manager is not running.
101	A command that cannot be executed at the same time as this command is executing.
102	You do not have the privileges required to execute this command.
110	A write error has occurred.
120	There is not enough free disk space for the media library.
200	Setup has not been performed.
255	A system error has occurred.

Command location

Admin server

[Windows]

<*Systemwalker Software Configuration Manager installation directory*>\SWCFMGM\bin

[Linux]

/opt/FJSVcfmgm/bin

Privilege required/execution environment

[Windows]

- Administrator privileges are required. If using Windows Server 2008 or later as your operating system, execute this command as an administrator.
- This command can be executed on the admin server.

[Linux]

- This command can only be executed by the system administrator (superuser).
- This command can be executed on the admin server.

Notes

Common notes

- This command cannot be executed simultaneously with other commands. An exclusive control error will occur if a command is executed while another command is still executing.
- Execute this command while Systemwalker Software Configuration Manager is running.

Notes regarding update

- The method cannot be changed if the software and a parameter setting definition are associated or a script package is registered.

Notes regarding deletion

- Use this command to delete information about parameter settings definitions that no longer need to be managed by Systemwalker Software Configuration Manager.
- Before deleting a parameter settings definition, ensure that there are no associated predefined parameters. The parameter settings definition cannot be deleted if such information exists.

Notes regarding list display

- If no parameter settings definitions have been registered, only the header information will be output.

#Definition ID,Product predefined,Definition name,Method,Package name

Notes regarding additions and update errors

- If a parameter settings definition file is invalid, the registration and update of the parameter settings definitions will be canceled and the following error message will be output:

An error occurred during XML file validation. [Details:<*Details* >]

Examples

- Add

  [Windows]

```
swcfmg_param_settingdef.exe -add -file C:\work\paramsettingdef.xml
```

[Linux]

```
swcfmg_param_settingdef -add -file /tmp/paramsettingdef.xml
```

- Update

[Windows]

```
swcfmg_param_settingdef.exe -update -id SS00000001 -file C:\work\paramsettingdef.xml
```

[Linux]

```
swcfmg_param_settingdef -update -id SS00000001 -file /tmp/paramsettingdef.xml
```

- Delete

[Windows]

```
swcfmg_param_settingdef.exe -delete -id SS00000001
```

[Linux]

```
swcfmg_param_settingdef -delete -id SS00000001
```

- List

[Windows]

```
swcfmg_param_settingdef.exe -list
```

[Linux]

```
swcfmg_param_settingdef -list
```

- Acquire

[Windows]

```
swcfmg_param_settingdef.exe -get -id SS00000001 -file C:\work\paramsettingdef.xml
```

[Linux]

```
swcfmg_param_settingdef -get -id SS00000001 -file /tmp/paramsettingdef.xml
```

- Upload

[Windows]

```
swcfmg_param_settingdef.exe -upload -id SS00000001 -file C:\work\paramsettingdef.zip
```

[Linux]

```
swcfmg_param_settingdef -upload -id SS00000001 -file /tmp/paramsettingdef.zip
```

- Download

[Windows]

```
swcfmg_param_settingdef.exe -download -id SS00000001 -dir C:\work\paramsettingdef
```

[Linux]

```
swcfmg_param_settingdef -download -id SS00000001 -dir /tmp/paramsettingdef
```

3.3.2 swcfmg_param_collectingdef (Parameter Collection Definition Management Command)

Description

Adds, updates, deletes, lists, and acquires parameter collection definitions. Also, uploads or downloads the script package associated with the script package of a parameter collection definition.

Uses a parameter collection definition input file as the input file for adding, updating, and acquiring. The parameter collection definition input file is in XML format. Refer to "2.2 Definition of Parameters to be Collected" for details.

Uses a script package for uploading and downloading. The script package is a zip file. Refer to "2.2.3 Package Files" for details.

Synopsis

[Windows]

```
swcfmg_param_collectingdef.exe
{ -add -file <Parameter collection definition file path> |
-update -id <Parameter collection definition ID> -file <Parameter collection definition file path>|
-delete -id <Parameter collection definition ID>,... |
-list [-file <Output file path>] [-e <Character encoding>] |
-get -id <Parameter collection definition ID> -file <Output file path> |
-upload -id <Parameter collection definition ID> -file <Script package file path> |
-download -id <Parameter collection definition ID> -dir <Output directory path> }
```

[Linux]

```
swcfmg_param_collectingdef
{ -add -file <Parameter collection definition file path> |
-update -id <Parameter collection definition ID> -file <Parameter collection definition file path>|
-delete -id <Parameter collection definition ID>,... |
-list [-file <Output file path>] [-e <Character encoding>] |
-get -id <Parameter collection definition ID> -file <Output file path> |
-upload -id <Parameter collection definition ID> -file <Script package file path> |
-download -id <Parameter collection definition ID> -dir <Output directory path> }
```

Options

-add -file *<Parameter collection definition file path>*

Adds the parameter collection definition using the specified parameter collection definition file. The input file is a parameter collection definition XML file.

A parameter collection definition ID is generated for the newly added parameter collection definition. Outputs this ID to standard output.

- Format of ID to be generated

  The parameter collection definition ID is automatically assigned a number according to the following rule:

```
SC (prefix) + 8-digit number
Example:
SC00000001
```

- Output format for standard output

  The parameter collection definition ID is output in the following format:

```
[<Parameter collection definition ID>]
```

-update -id <*Parameter collection definition* ID> -file <*Parameter collection definition file path*>

Updates the parameter collection definition for the parameter collection definition ID using the specified parameter collection definition file. The input file is a parameter collection definition XML file.

-delete -id <*Parameter collection definition ID*>,

Deletes the parameter collection definition from the parameter collection definition ID. Also, deletes the association (if any) between the software and the parameter collection definition. Multiple parameter collection definition IDs can be specified by delimiting them with "," (comma).

-list [-file <*Output file path*>]

Uses a parameter collection definition list file in CSV format to output a list of registered parameter collection definitions.

If -file <*Output file path*> is omitted, information will be output to the standard output for the command.

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If output to the file fails due to an error, no new file will be created. If an existing file was to be overwritten, it will be kept as is without being overwritten.

- Output format for standard output

  Outputs the list of parameter collection definitions in the following format:

Item No.	Item	Remarks
1	Parameter collection definition ID	Outputs the parameter collection definition ID.
2	Is the parameter collection definition product-predefined?	Outputs whether the parameter collection definition is product-predefined.
3	Parameter collection definition name	Outputs the parameter collection definition name.
4	Parameter collection method	Outputs the parameter collection method.
3	Script package file name	Outputs the name of the script package file.

- Header format

  Insert the header in the first line of the CSV file using the format shown below. The header is output even if there is no information to be output to the file.

```
#Definition ID,Product predefined,Definition name,Method,Package name
```

-get -id <*Parameter collection definition ID*> -file <*Output file path*>

Outputs the parameter collection definition for the parameter collection definition ID to the specified output file path. The output file is a parameter collection definition XML file.

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If output to the file fails due to an error, no new file will be created. If an existing file was to be overwritten, it will be kept as is without being overwritten.

-upload -id <*Parameter collection definition ID*> -file <*Script package file path*>

Uploads the script package to the parameter collection definition for the parameter collection definition ID. The input file is a script package zip file.

Script packages that can be uploaded are subject to the following conditions. An error will occur if conditions other than these are used.

- File extension must be .zip.

  Ensure that the file extension is .zip.

- File size must be no greater than 2 MB.

  The maximum uploadable file size is 2 MB.

- Store the discovery script in the root.

  Store the discovery script (discover.cmd or discover.sh) in the root of the zip file rather than under the directory.

-download -id <*Parameter collection definition ID*> -dir <*Output directory path*>

Downloads the script package registered in the parameter collection definition of the parameter collection definition ID. Outputs script package zip files to the output directory path.

Ensure that no files exist in the output directory, or an error will occur.

-e <*Character encoding*>

Specify the character encoding for the file.

The character encodings below can be specified (single-byte uppercase characters must be used):

- UTF-8

If this option is omitted, the character encoding of the execution environment of this command will be used.

Return value

Return value	Meaning
0	Completed normally.
10	A parameter error has occurred.
20	The specified I/O path is incorrect.
30	The specified character encoding is incorrect.
40	Unable to analyze the file.
50	The file description is incorrect.
70	The specified ID does not exist.
71	The specified ID is product-predefined.
72	The package has not been registered in the specified ID.
73	The package has not been correctly registered in the specified ID.
100	Systemwalker Software Configuration Manager is not running.
101	A command that cannot be executed at the same time as this command is executing.
102	You do not have the privileges required to execute this command.
110	A write error has occurred.
120	There is not enough free disk space for the media library.
200	Setup has not been performed.
255	A system error has occurred.

Command location

Admin server

[Windows]

```
<Systemwalker Software Configuration Manager installation directory>\SWCFMGM\bin
```

[Linux]

```
/opt/FJSVcfmgm/bin
```

Privilege required/execution environment

[Windows]

- Administrator privileges are required. If using Windows Server 2008 or later as your operating system, execute this command as an administrator.
- This command can be executed on the admin server.

[Linux]

- This command can only be executed by the system administrator (superuser).
- This command can be executed on the admin server.

Notes

Common notes

- This command cannot be executed simultaneously with other commands. An exclusive control error will occur if a command is executed while another command is still executing.
- Execute this command while Systemwalker Software Configuration Manager is running.

Notes regarding update

- The method cannot be changed if the software and a parameter collection definition are associated or a script package is registered.
- If multiple parameter collection definitions are associated with a single software product, you cannot specify a duplicate key for the parameter list.

Notes regarding deletion

- Use this command to delete information about parameter collection definitions that no longer need to be managed by Systemwalker Software Configuration Manager.

Notes regarding list display

- If no parameter collection definitions have been registered, only the header information will be output.

```
#Definition ID,Product predefined,Definition name,Method,Package name
```

Notes regarding additions and update errors

- If a parameter collection definition file is invalid, the registration and update of the parameter collection definitions will be canceled and the following error message will be output:

```
An error occurred during XML file validation.[Details:<Cause>]
```

Examples

- Add

  [Windows]

```
swcfmg_param_collectingdef.exe -add -file C:\work\paramcollectingdef.xml
```

[Linux]

```
swcfmg_param_collectingdef -add -file /tmp/paramcollectingdef.xml
```

- Update

[Windows]

```
swcfmg_param_collectingdef.exe -update -id SC00000001 -file C:\work\paramcollectingdef.xml
```

[Linux]

```
swcfmg_param_collectingdef -update -id SC00000001 -file /tmp/paramcollectingdef.xml
```

- Delete

[Windows]

```
swcfmg_param_collectingdef.exe -delete -id SC00000001
```

[Linux]

```
swcfmg_param_collectingdef -delete -id SC00000001
```

- List

[Windows]

```
swcfmg_param_collectingdef.exe -list
```

[Linux]

```
swcfmg_param_collectingdef -list
```

- Acquire

[Windows]

```
swcfmg_param_collectingdef.exe -get -id SC00000001 -file C:\work\paramcollectingdef.xml
```

[Linux]

```
swcfmg_param_collectingdef -get -id SC00000001 -file /tmp/paramcollectingdef.xml
```

- Upload

[Windows]

```
swcfmg_param_collectingdef.exe -upload -id SC00000001 -file C:\work\paramcollectingdef.zip
```

[Linux]

```
swcfmg_param_collectingdef -upload -id SC00000001 -file /tmp/paramcollectingdef.zip
```

- Download

[Windows]

```
swcfmg_param_collectingdef.exe -download -id SC00000001 -dir C:\work\paramcollectingdef
```

[Linux]

```
swcfmg_param_collectingdef -download -id SC00000001 -dir /tmp/paramcollectingdef
```

3.3.3 swcfmg_param_defassoc (Command to Associate Software and Parameter Definitions)

Description

Associates or disassociates the software settings definition with the parameter settings definition, or the software collection definition with the parameter collection definition, and displays them in list form.

Synopsis

[Windows]

```
swcfmg_param_defassoc.exe
{ -attach -softid <Software ID> -sdefid <Parameter settings definition ID> |
-attach -softid <Software ID> -cdefid <Parameter collection definition ID> |
-detach -softid <Software ID> -sdefid <Parameter settings definition ID> |
-detach -softid < Software ID > -cdefid <Parameter collection definition ID> |
-list [-sdef] [-cdef] [-file <Output file path>] [-e <Input file character encoding>] }
```

[Linux]

```
swcfmg_param_defassoc
{ -attach -softid <Software ID> -sdefid <Parameter settings definition ID> |
-attach -softid <Software ID> -cdefid <Parameter collection definition ID> |
-detach -softid <Software ID> -sdefid <Parameter settings definition ID> |
-detach -softid < Software ID > -cdefid <Parameter collection definition ID> |
-list [-sdef] [-cdef] [-file <Output file path>] [-e <Input file character encoding>] }
```

Options

-attach -softid <*Software ID*> -sdefid <*Parameter settings definition ID*>

Associates the software that has the specified ID and the parameter settings definition that has the specified ID. Note that multiple parameter settings definitions can be associated with a single software product. A parameter definition method compatible with the OS type of the software must be specified.

-attach -softid <*Software ID*> -cdefid <*Parameter collection definition ID*>

Associates the software that has the specified ID and the parameter collection definition that has the specified ID. Note that a single parameter collection definition can be associated with multiple software products. A parameter definition method compatible with the OS type of the software must be specified.

-detach -softid <*Software ID*> -sdefid <*Parameter settings definition ID*>

Removes the association between the software that has the specified ID and the parameter settings definition that has the specified ID

-detach -softid <*Software ID*> -cdefid <*Parameter collection definition ID*>

Removes the association between the software that has the specified ID and the parameter collection definition that has the specified ID

-list [-sdef] [-cdef] [-file <*Output file path*>]

Outputs a list of associations in CSV format

-sdef Outputs a list of only those associations between the software and parameter settings definitions

-cdef Outputs a list of only those associations between the software and parameter collection definitions

If the specified file does not exist, a new file will be created. If the file exists, the existing file will be overwritten.

If output to the file fails due to an error, no new file will be created. If an existing file was to be overwritten, it will be kept as is without being overwritten.

- Output format for standard output

  Outputs the list of associations in the following format:

Item No.	Item	Remarks
1	Software ID	Outputs the software ID.
2	Definition ID	Outputs the parameter settings ID or the parameter collection ID.
3	Software name	Outputs the software product name.
4	Version	Outputs the software version.
5	OS type	Outputs the software OS.
6	Definition name	Outputs the parameter settings definition name or the parameter collection definition name.

- Header format

  Insert the header in the first line of the CSV file using the format shown below. The header is output even if there is no information to be output to the file.

```
#Software ID,Definition ID,Software name,Version,Vendor,Definition name
```

-e <*Character encoding*>

Specify the character encoding for the file.

The character encodings below can be specified (single-byte uppercase characters must be used):

- UTF-8

If this option is omitted, the character encoding of the execution environment of this command will be used.

Return value

Return value	Meaning
0	Completed normally.
10	A parameter error has occurred.
20	The specified I/O path is incorrect.
30	The specified character encoding is incorrect.
70	The specified ID does not exist.
71	The software and parameter definition are already associated.
100	Systemwalker Software Configuration Manager is not running.
101	A command that cannot be executed at the same time as this command is executing.
102	You do not have the privileges required to execute this command.
110	A write error has occurred.
200	Setup has not been performed.
255	A system error has occurred.

Command location

Admin server

[Windows]

```
<Systemwalker Software Configuration Manager installation directory>\SWCFMGM\bin
```

[Linux]

```
/opt/FJSVcfmgm/bin
```

Privilege required/execution environment

[Windows]

- Administrator privileges are required. If using Windows Server 2008 or later as your operating system, execute this command as an administrator.
- This command can be executed on the admin server.

[Linux]

- This command can only be executed by the system administrator (superuser).
- This command can be executed on the admin server.

Notes

Common notes

- This command cannot be executed simultaneously with other commands. An exclusive control error will occur if a command is executed while another command is still executing.
- Execute this command while Systemwalker Software Configuration Manager is running.

Notes regarding association

- If multiple parameter collection definitions are associated with a single software product, you cannot specify a duplicate key for the parameter list.
- When the software type is Solaris, that software cannot be associated with the parameter definitions.

Notes regarding removal of associations

- Use this command to remove associations that no longer need to be managed by Systemwalker Software Configuration Manager.

Notes regarding list display

- If no predefined parameters have been registered, only the header information will be output.

```
#Software ID,Definition ID,Software name,Version,Vendor,Definition name
```

Examples

- Associate

[Windows]

```
swcfmg_param_defassoc.exe -attach -softid UDP00001 -sdefid SS00000001
```

[Linux]

```
swcfmg_param_defassoc -attach -softid UDP00001 -sdefid SS00000001
```

- Disassociate

[Windows]

```
swcfmg_param_defassoc.exe -detach -softid UDP00001 -sdefid SS00000001
```

[Linux]

```
swcfmg_param_defassoc -detach -softid UDP00001 -sdefid SS00000001
```

- List

[Windows]

```
swcfmg_param_defassoc.exe -list
```

[Linux]

```
swcfmg_param_defassoc -list
```

Chapter 4 File Reference

4.1 File List

The following table lists the definition files.

Type	Feature name	Overview
Software information definition files	Software information file	Registers and updates software information.
	Installed software information file	Registers and deletes installed software information.
Parameter information definition files	Parameter settings definition file	Defines configuration information for parameters that can be set in the software.
	Parameter information file	Defines the values to be set in the software parameters.
	Parameter collection definition file	Defines configuration information for parameters to be collected by the software.

4.2 Software Information Definition Files

This section explains the software information definition files.

4.2.1 Software Information File

File name

Any name

Description

This file is used to register and update software information.

Register and update software information using the swcfmg_software (Software Information Management Command) options.

File location

Any folder

File format

CSV format

[#][*<software-id>*],*<software-name>*,*<version>*,*<os-type>*,*<vendor>*

...

Parameters

The following is regarded as a single definition:

[#][<software-id>],<software-name>,<version>,<os-type>,<vendor>

Column	Item	Meaning	Value	Remarks
	#	Comment line		

Column	Item	Meaning	Value	Remarks
1	software-id	Software ID	Example: UDP00001	Can be omitted. UDP followed by a five-digit number The software ID is automatically collected during registration (do not configure this column during registration). Specify the ID output in the list displayed during update.
2	software-name	Software name	Example: ABCDE-Software	Cannot be omitted.
3	version	Software version	Example: 1.0	Cannot be omitted.
4	os-type	OS type	Example: Windows	Cannot be omitted.
5	vendor	Software vendor name	Example: ABCDE Company	Cannot be omitted.

Notes

None

Examples

- During registration

```
,ABCDE-Software,1.0,Windows,ABCDE Company
,FGHIJ-Software,2.0,Windows, ABCDE Company
```

- During update

```
UDP00001,ABCDE-Software,1.1,Windows,ABCDE Company
UDP00002,FGHIJ-Software,2.0.1,Windows, ABCDE Company
```

4.2.2 Installed Software Information File

File name

Any name

Description

This file is used to register and delete installed software information.

Register and update installed software information using the swcfmg_installed software (Installed Software Information Management Command) options.

File location

Any folder

File format

CSV format

```
[#]<ip-address>,<software-id>
...
```

Parameters

The following constitutes a single definition:

[#]<ip-address>,<software-id>

Column	Item	Meaning	Value	Remarks
	#	Comment line		
1	ip-address	Management IP address of the managed server	Example: 10.10.10.10	Cannot be omitted.
2	software-id	Software ID	Example: UDP00001	Cannot be omitted. UDP followed by a five-digit number Specify the software ID automatically assigned when the new software information was registered using the Software Information Management Command (swcfmg_software).

Notes

None

Examples

```
10.10.10.11,UDP000001
10.10.10.11,UDP000002
10.10.10.12,UDP000001
```

4.3 Parameter Information Definition Files

This section describes the parameter information definition files.

4.3.1 Parameter Settings Definition File

File name

Any name

Description

This is an XML file that describes the configuration information for parameters that can be set in the software.

Refer to "2.1 Definition of Parameters to be Set" for information on parameter settings definitions.

File format

Parameter settings definition XML files will have the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterSetting version="1.0">
    <name>parmSettingsDefinitionName</name>
    <description>desc</description>
    <method>parmSetupMethod</method>
    <parameters>
        <parameter>
            <key>[parmKey]</key>
            <type>[parmValType]</type>
            <value>[defaultValIfParmIsMandatory (if type is boolean, number, or string)]</value>
            <array>[defaultValIfParmIsMandatory (if type is string array)]
                <element>elementVal</element>
                ...
            </array>
            <map>[defaultValIfParmIsMandatory> (if type is map)]
                <entry>
                    <subkey>mapSubkey</subkey>
                    <value>mapVal</value>
                </entry>
                ...
            </map>
            <label>[parmLabel]</label>
            <description>[parmDesc]</description>
        </parameter>
        ...
    </parameters>
</parameterSetting>
```

The table below describes the items (tags) and their settings.

Modify parameter settings definitions as necessary, based on the information in this table.

Tag names in square brackets [ ] are optional.

Tag name	Type	Allowable range	Description	Mandatory	Settings
name	string UTF-8	256 characters or less	Specifies the parameter settings definition name.	O	
description	string UTF-8	256 characters or less	Specifies the parameter settings definition description.	O	
method	string ASCII	Select an option	Specifies the parameter setup method.	Y	Select from the following options: - "cmd": Calls the startup script (startup.cmd). - "sh": Calls the startup script (startup.sh).
parameters	-	-	Specifies multiple parameters that can be configured in the software.	N	
parameter	-	1 or more	Specifies the parameter key and value pairs that can	N	

Tag name	Type	Allowable range	Description	Mandatory	Settings
			be configured in the software.		
key	string ASCII	256 bytes or less	Specifies the parameter key.	Y	Characters that can be used are alphanumeric characters, ".", "_", and "-". However, the first character must only be alphabetic.
type	string ASCII	Select an option	Specifies the value type.	Y	Select from the following options: - boolean: true, false - number: Number - string: Character string - string array: Character string array - map Refer to "2.1.2 Parameter Types" for information on types.
[value]	string UTF-8	4096 characters or less	Specifies the default value if a value must be set. The value can be changed when configuring this parameter, but it cannot be left blank. Can only be specified if type is boolean, number or string. The <value>, <array> and <map> tags are mutually exclusive.	O	The <type> tag determines what type of value can be specified. Refer to "2.1 Definition of Parameters to be Set" for information on values.
[array]	-	-	Specifies the default value if a value must be set. Can only be specified if the type is string array. The <value>, <array> and <map> tags are mutually exclusive.	N	Use the element tag to specify array elements. Do not specify the element tag if there are zero arrays.
[element]	string UTF-8	4096 characters or less	Specifies the elements in an array when the default	Y	Use a character string to specify the elements in an array. It is also

Tag name	Type	Allowable range	Description	Mandatory	Settings
			value type for the mandatory parameter is "array".		possible to specify an empty string.
[map]	-	-	Specifies the default value if a value must be set. Can only be specified if type is map. The <value>, <array> and <map> tags are mutually exclusive.	N	Use the entry tag to specify map entries. Do not specify the entry tag if there are zero maps.
[entry]	-	-	Used with each key-value pair to specify the value when the default value type for the mandatory parameter is map.	N	Use the key tag to specify keys and the value tag to specify values.
subkey	string ASCII	256 bytes or less	Specifies the subkey when the default value type for the mandatory parameter is map.	Y	Characters that can be used are alphanumeric characters, periods (.), underscores (_), and hyphens (-). However, the first character must only be alphabetic.
value	string UTF-8	4096 characters or less	Specifies the value when the default value type for the mandatory parameter is map.	Y	Use a character string to specify map values. It is also possible to specify an empty string.
[label]	string UTF-8	64 characters or less	Specifies the label used to display the parameter in the window.	O	
[description]	string UTF-8	256 characters or less	Specifies the parameter description.	O	

Legend for column 'Mandatory':

Y: If the tag is specified, then a value is also required

O: Optional

N: The tag can be specified without a value

Example (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterSetting version="1.0">
    <name>Setting Definition</name>
    <description>Parameter Setting Definition</description>
    <method>cmd</method>
    <parameters>
```

```
        <parameter>
            <key>key.isParameter</key>
            <type>boolean</type>
            <value>true</value>
            <label>isParameter</label>
            <description>boolean</description>
        </parameter>
        <parameter>
            <key>key.number</key>
            <type>number</type>
            <value>100</value>
            <label>number</label>
            <description>number</description>
        </parameter>
        <parameter>
            <key>key.parameter</key>
            <type>string</type>
            <value>parameter</value>
            <label>parameter</label>
            <description>string</description>
        </parameter>
        <parameter>
            <key>key.parameter.list</key>
            <type>string array</type>
            <array>
                <element>first</element>
                <element>second</element>
            </array>
            <label>list</label>
            <description>array</description>
        </parameter>
        <parameter>
            <key>key.parameters</key>
            <type>map</type>
            <map>
                <entry>
                    <subkey>subkey.right</subkey>
                    <value>right</value>
                </entry>
                <entry>
                    <subkey>subkey.left</subkey>
                    <value>left</value>
                </entry>
            </map>
            <label>parameters</label>
            <description>map</description>
        </parameter>
    </parameters>
</parameterSetting>
```

4.3.2 Parameter Information File

File name

Any name

Description

This is a file in CSV or XML format that describes the values to be set in the software parameters. Parameters for which values can be set in the software are the parameters defined in the parameter settings definition.

Refer to "Parameter Value Settings" in the *Operation Guide* for details on parameter information.

CSV file format

Parameter information CSV files will have the format shown below.

Modify the parameter information as necessary, based on this information.

Purpose of line	Line/row	Row 1	Row 2	Row 3	Row 4
Header	Line 1	#Parameters	Key	Value	Status
Name	Line 2	parameterInfo	name	[Parameter information name]	
Description	Line 3	parameterInfo	description	[Parameter information description]	
Parameters	Line 4	parameters	[Parameter key]	[Parameter value]	[Parameter status]
...	Line 5...	parameters	...	...	...

Explanation of lines

The following table explains the lines and their settings:

Purpose of line	Description	Mandatory	Settings
Header	The header is deemed to be specified if the first line of the CSV file starts with a hash (#) symbol.	N	Apart from the # symbol, the header may contain any content. Omit the line if not specifying a header.
Name	This line specifies the parameter information name.	Y	
Description	This line specifies the parameter information description.	N	Omit the line if not specifying a parameter information description.
Parameters	This line specifies the parameter key and value.	N	Add the same number of lines as the number of parameter keys you wish to specify. Omit the line if not specifying any parameter keys.

Explanation of items

The following table explains the item names and their settings.

Item names in square brackets [ ] can be omitted line by line. Omit the fourth row if omitting all parameter statuses.

Item	Type	Allowable range	Description	Mandatory	Settings
Parameter information name	string	256 characters or less	Specifies the parameter information name.	N	
[Parameter information description]	string	256 characters or less	Specifies the description of the parameter information.	N	
Parameter key	string	If type is boolean, number or string:	If type is boolean, number of string: specifies the parameter key.	Y	Only the key contained in the parameter settings definition can be specified.

Item	Type	Allowable range	Description	Mandatory	Settings
		256 bytes or less If type is string array or map: 514 bytes or less	If type is string array: specifies the element index. If type is map: specifies the subkey.		Enclose the string array index and the map subkey in square brackets ([ ]). The string array index starts from 1. When using a parameter settings definition from "Middleware Parameter Settings Information", specify the parameter key.
Parameter value	string	4096 characters or less	Specifies the parameter value.	N	The value depends on the parameter settings definition restrictions. Refer to "Predefined Parameters" of the *Operation Guide* for information on values.
Parameter status	string	32 characters or less	Specifies the parameter value.	N	Specify whether to set a parameter or not. The following values can be set to indicate the status: - Set a value: set - Do not set a value: Empty string, notset

Key-value description

Keys and values should be specified as shown in the table below:

Parameters		Specification	
Type	Value characteristics	Key	Value
boolean	Value specified	Key	Value
number	Value specified	Key	Value
string	Value specified	Key	Value
	Empty string	Key	Empty string
string array	Value specified (1 or more elements)	Key [index]	Value
	Empty string (1 or more elements)	Key [index]	Empty string
	0 elements	Key	Empty string
map	Value specified (1 or more entries)	Key [subkey]	Value
	Empty string (1 or more entries)	Key [subkey]	Empty string
	0 entries	Key	Empty string

Example (CSV)

```
#Parameters,Key,Value,Status
parameterInfo,name,Parameter Info,
parameterInfo,description,Parameter Info,
parameters,key.isParameter,true,set
parameters,key.number,100,set
parameters,key.parameter,parameter,set
parameters,key.parameter.list[1],first,set
parameters,key.parameter.list[2],second,set
parameters,key.parameter.list[subkey.right],right,set
parameters,key.parameter.list[subkey.left],left,set
```

XML file format

Parameter information XML files will have the format shown below.

Modify the parameter information as necessary, based on this information.

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterInfo version="3.0">
    <name><parm information name></name>
    <description><desc></description>
    <parameters>
        <parameter>
            <key>[<parm key>]</key>
            <value>[<parm val> (if type is boolean, number, or string)]</value>
            <array>[<parm val> (if type is string array)]
                <element><element val> </element>
                ...
            </array>
            <map>[<parm val> (if type is map)] *1 *4
                <entry>
                    <subkey><map subkey></subkey>
                    <value><map val></value>
                </entry>
                ...
            </map>
        </parameter>
        ...
    </parameters>
</parameterInfo>
```

The table below describes the items (tags) and their settings.

Tag names in square brackets [ ] are optional.

Tag name	Type	Allowable range	Description	Mandatory	Settings
name	string UTF-8	256 characters or less	Specifies the parameter information name.	O	
[description]	string UTF-8	256 characters or less	Specifies the description of the parameter information.	O	
[parameters]	-	-	Specifies multiple parameters to be set in the software.	N	
parameter	-	0 or more	Specifies the parameter key and value pairs to be set in the software.	N	
key	string ASCII	256 bytes or less	Specifies the parameter key.	Y	Only the key contained in the

Tag name	Type	Allowable range	Description	Mandatory	Settings
					parameter settings definition can be specified. When using a parameter settings definition from "Middleware Parameter Settings Information", specify the parameter key.
[value]	string UTF-8	4096 characters or less	Specifies the parameter value. Can only be specified if type is boolean, number or string. One and only one <value>, <array>, or <map> tag must be specified.	O	The value depends on the parameter settings definition restrictions. Refer to "Predefined Parameters" of *Operation Guide* for information on values.
[array]	-	-	Specifies the parameter value. Can only be specified if type is string array. One and only one <value>, <array>, or <map> tag must be specified.	N	Use the element tag to specify array elements. Do not specify the element tag if there are zero arrays.
[element]	string UTF-8	4096 characters or less	Specifies the elements in an array when the parameter value type is "array".	Y	Use a character string to specify the elements in an array. It is also possible to specify an empty string.
[map]	-	-	Specifies the parameter value. Can only be specified if type is map. One and only one <value>, <array>, or <map> tag must be specified.	N	Use the entry tag to specify map entries. Do not specify the entry tag if there are zero maps.
[entry]	-	-	Used with each key-value pair to specify the value when the parameter value type is map.	N	Use the key tag to specify keys and the value tag to specify values.
subkey	string ASCII	256 bytes or less	Specifies the subkey when the parameter value type is map.	Y	Characters that can be used are alphanumeric characters, periods (.), underscores (_), and hyphens (-). However, the first

Tag name	Type	Allowable range	Description	Mandatory	Settings
					character must only be alphabetic.
value	string UTF-8	4096 characters or less	Specifies the value when the parameter value type is map.	Y	Use a character string to specify map values. It is also possible to specify an empty string.

Legend for column 'Mandatory':

Y: If the tag is specified, then a value is also required

O: Optional

N: The tag can be specified without a value

Example (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterInfo version="3.0">
    <name>Parameter Info</name>
    <description>Parameter Info</description>
    <parameters>
        <parameter>
            <key>key.isParameter</key>
            <value>true</value>
        </parameter>
        <parameter>
            <key>key.number</key>
            <value>100</value>
        </parameter>
        <parameter>
            <key>key.parameter</key>
            <value>parameter</value>
        </parameter>
        <parameter>
            <key>key.parameter.list</key>
            <array>
                <element>first</element>
                <element>second</element>
            </array>
        </parameter>
        <parameter>
            <key>key.parameters</key>
            <map>
                <entry>
                    <subkey>subkey.right</subkey>
                    <value>right</value>
                </entry>
                <entry>
                    <subkey>subkey.left</subkey>
                    <value>left</value>
                </entry>
            </map>
        </parameter>
    </parameters>
</parameterInfo>
```

4.3.3 Parameter Collection Definition File

File name

Any name

Description

This is an XML file that describes the configuration information for parameters to be collected by the software.

Refer to "2.2 Definition of Parameters to be Collected" for information on parameter collection definitions.

File format

Parameter collection definition XML files will have the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterCollecting version="1.0">
    <name>parameterCollectionName</name>
    <method>parameterCollectionMethod</method>
    <parameters>
        <parameter>
            <key>[parameterKey]</key>
            <type>[parameterValueType]</type>
            <label>[parameterLabel]</label>
        </parameter>
        ...
    </parameters>
</parameterCollecting>
```

The table below describes the items (tags) and their settings.

Modify parameter settings definitions as necessary, based on the information in this table.

Tag names in square brackets [ ] are optional.

Tag name	Type	Allowable range	Description	Mandatory	Settings
name	string UTF-8	256 characters or less	Specifies the parameter collection name.	O	
method	string ASCII	Select an option	Specifies the parameter collection method.	Y	Select from the following options: - "cmd": Calls the discovery script (discover.cmd) - "sh": Calls the discovery script (discover.sh)
parameters	-	-	Specifies multiple parameters that can be collected from the software.	N	
parameter	-	1 or more	Specifies the parameter key and value pairs that can be collected from the software.	N	
key	string ASCII	256 bytes or less	Specifies the parameter key.	Y	Characters that can be used are alphanumeric characters, ".", "_", and "-".

Tag name	Type	Allowable range	Description	Mandatory	Settings
					However, the first character must only be alphabetic.
type	string ASCII	Select an option	Specifies the value type.	Y	Select from the following options: - boolean - number - string - string array - map Refer to "2.1.2 Parameter Types" for information on types.
[label]	string UTF-8	64 characters or less	Specifies the label used to display the parameter in the window.	O	

Legend for column 'Mandatory':

Y: If the tag is specified, then a value is also required

O: Optional

N: The tag can be specified without a value

Example (XML)

```
<?xml version="1.0" encoding="UTF-8"?>
<parameterCollecting version="1.0">
    <name>Collecting Definition</name>
    <method>cmd</method>
    <parameters>
        <parameter>
            <key>key.isParameter</key>
            <type>boolean</type>
        </parameter>
        <parameter>
            <key>key.number</key>
            <type>number</type>
        </parameter>
        <parameter>
            <key>key.parameter</key>
            <type>string</type>
        </parameter>
        <parameter>
            <key>key.parameter.list</key>
            <type>string array</type>
        </parameter>
        <parameter>
            <key>key.parameters</key>
            <type>map</type>
        </parameter>
    </parameters>
</parameterCollecting>
```

Chapter 5 Script Reference

This chapter explains the scripts provided by Systemwalker Software Configuration Manager.

5.1 Script List

The following table lists the scripts used for managing parameters.

Type	Feature name	Overview
Parameter settings script	Startup script	Configures the software parameters.
Parameter collection script	Discovery script	Collects parameter information from the software and outputs it to a file.

5.2 Parameter Settings Script

This section explains the parameter settings script.

5.2.1 Startup Script

Synopsis

Startup

Description

Configures software parameters.

Output format

"0" is returned if the script ends normally, and some other value is returned otherwise - error messages are sent either to the standard output or to the standard error output.

Notes

The directory in which the command is stored will be considered the current directory.

The startup script location and the files that can be referenced by it are displayed below:

```
<work directory>
+ <software ID>
  + scriptpkg
  |  + Startup script(startup.cmd, startup.sh)
  |  + Environment variable setup script(setenv.cmd, setenv.sh)
  |  + Script package file attachments
  + paramdata
  |  +XML file with parameter information to be set(parameterinfo.xml)
  + parampkg
      + Parameter package file attachments
```

5.3 Parameter Collection Script

This section explains the parameter collection script.

5.3.1 Discovery Script

Synopsis

```
discover -dir <output dir>
```

Description

Collects parameter information from the software and outputs it to a file.

Options

-dir

Specify the output directory for the parameter information XML file (parameterinfo.xml) and file attachments.

If the path contains spaces, enclose it in double quotation marks (").

Output format

"0" is returned if the script ends normally, and some other value is returned otherwise - error messages are sent either to the standard output or to the standard error output. The output is sent to the directory below, in the format below.

```
+ <output directory>
  + XML file with information about parameters collected (parameterinfo.xml)
  + File attachment with collected parameters (includes those that will be output and those that will
not)
```

Notes

The directory which stores the command will be used as the current directory.

An error will occur if the directory does not exist, or if it exists and contains files or directories.