



# Opsware<sup>®</sup> SAS Web Services API 2.2 Guide

**Corporate Headquarters**

---

599 North Mathilda Avenue Sunnyvale, California 94085 U.S.A.  
T + 1 408.744.7300 F +1 408.744.7383 [www.opsware.com](http://www.opsware.com)

Copyright © 2000-2005 Opsware Inc. All Rights Reserved.

Opsware Inc. Unpublished Confidential Information. NOT for Redistribution. All Rights Reserved.

Opsware is protected by U.S. Patent Nos. 6,658,426, 6,751,702, 6,816,897, 6,763,361 and patents pending

Opsware, Opsware Command Center, Model Repository, Data Access Engine, Web Services Data Access Engine, Software Repository, Command Engine, Opsware Agent, Model Repository Multimaster Component, and Code Deployment & Rollback are trademarks and service marks of Opsware Inc. All other marks mentioned in this document are the property of their respective owners.

Additional proprietary information about third party and open source materials can be found at <http://www.opsware.com/support/opensourcedoc.pdf>.

# Table Of Contents

<b>Preface</b>	<b>xi</b>
<b>About this Guide</b> .....	<b>xi</b>
Audience for this Guide .....	xi
Contents of this Guide .....	xi
Conventions in this Guide .....	xii
Icons in this Guide .....	xiii
<b>Documentation for the Opware SAS Web Services API 2.2</b> .....	<b>xiv</b>
<b>How to Read this Guide</b> .....	<b>xiv</b>
Web Service Operations and WSDL Files .....	xv
Complex Data Types and WSDL Files .....	xvi
Array and List Complex Types .....	xvi
<b>Chapter 1: Overview</b>	<b>1</b>
<b>Overview of the Opware SAS Web Services API 2.2</b> .....	<b>1</b>
Overview of Features .....	1
What's New in Version 2.2 .....	2
Web Services in this Version .....	3
Version History .....	4
<b>Architecture of the Opware SAS Web Services API 2.2</b> .....	<b>5</b>
Client Architecture .....	6
WSDL Files and Web Services URLs .....	7
<b>Supported Client Languages and Technologies</b> .....	<b>8</b>

<b>About the Developer Syntax Docs</b> .....	<b>9</b>
Java Developer Docs (javadocs) .....	9
Python Developer Docs .....	9
Perl Developer Docs .....	10
C# Developer Docs .....	10
<b>Chapter 2: Setup</b> .....	<b>13</b>
<b>Setup Requirements</b> .....	<b>13</b>
Setup Requirements for All Client Languages .....	13
Java Client Setup .....	14
Python Client Setup .....	15
Perl Client Setup .....	16
C# Client Setup .....	17
<b>Authentication and Authorization</b> .....	<b>17</b>
HTTPS Access .....	18
<b>Exception Handling</b> .....	<b>18</b>
SOAP Fault Elements .....	18
Java Client Exceptions .....	19
Python Client Exceptions .....	20
Perl Client Exceptions .....	20
C# Client Exceptions .....	21
<b>Verbose Mode</b> .....	<b>22</b>
Perl Clients in Verbose Mode .....	22
Python Clients in Verbose Mode .....	22
<b>Logging</b> .....	<b>22</b>
<b>About the Sample Applications</b> .....	<b>23</b>

---

Sample Application Packages .....	23
Steps Performed by the Sample Client Applications.....	24
<b>Chapter 3: Command Session Web Service</b>	<b>27</b>
<hr/>	
<b>get.</b> .....	<b>29</b>
<b>getDSESessionCommandResult</b> .....	<b>30</b>
<b>getDSEServerProgress</b> .....	<b>31</b>
<b>getDSEServerResult</b> .....	<b>32</b>
<b>getDSESessionProgress</b> .....	<b>33</b>
<b>getDSESessionResult</b> .....	<b>34</b>
<b>getReconcileServerProgress</b> .....	<b>35</b>
<b>getReconcileServerResult</b> .....	<b>36</b>
<b>getReconcileServerResultPreview</b> .....	<b>37</b>
<b>getReconcileSessionProgress</b> .....	<b>38</b>
<b>getReconcileSessionResult</b> .....	<b>39</b>
<b>getReconcileSessionResultPreview</b> .....	<b>40</b>
<b>getStatus</b> .....	<b>41</b>
<b>Chapter 4: Custom Attribute Web Service</b>	<b>43</b>
<hr/>	
<b>get.</b> .....	<b>44</b>
<b>getKeyList</b> .....	<b>46</b>
<b>getList.</b> .....	<b>48</b>
<b>getValues</b> .....	<b>50</b>
<b>remove</b> .....	<b>51</b>
<b>set.</b> .....	<b>52</b>
<b>Chapter 5: Customer Web Service</b>	<b>53</b>
<hr/>	
<b>associateWithFacility</b> .....	<b>54</b>

<b>create</b> .....	<b>55</b>
<b>delete</b> .....	<b>57</b>
<b>get</b> .....	<b>59</b>
<b>getIDs</b> .....	<b>60</b>
<b>getList</b> .....	<b>62</b>
<b>update</b> .....	<b>63</b>
<b>Chapter 6: Distributed Script Web Service</b> .....	<b>65</b>

---

<b>create</b> .....	<b>66</b>
<b>delete</b> .....	<b>69</b>
<b>execute</b> .....	<b>70</b>
<b>get</b> .....	<b>72</b>
<b>getCurrentVersion</b> .....	<b>73</b>
<b>getScriptText</b> .....	<b>74</b>
<b>update</b> .....	<b>75</b>
<b>updateScriptText</b> .....	<b>77</b>
<b>Chapter 7: Facility Web Service</b> .....	<b>79</b>

---

<b>create</b> .....	<b>80</b>
<b>decommission</b> .....	<b>81</b>
<b>get</b> .....	<b>82</b>
<b>getIDs</b> .....	<b>83</b>
<b>getList</b> .....	<b>84</b>
<b>update</b> .....	<b>85</b>
<b>Chapter 8: Node Web Service</b> .....	<b>87</b>

---

<b>attachServer</b> .....	<b>89</b>
<b>create</b> .....	<b>90</b>

---

<b>delete</b> .....	<b>92</b>
<b>detachCustomers</b> .....	<b>93</b>
<b>detachPlatforms</b> .....	<b>94</b>
<b>detachServer</b> .....	<b>95</b>
<b>detachSoftwarePackage</b> .....	<b>96</b>
<b>get.</b> .....	<b>97</b>
<b>getChild</b> .....	<b>98</b>
<b>getChildList</b> .....	<b>99</b>
<b>getCustomerList</b> .....	<b>100</b>
<b>getDescendant</b> .....	<b>101</b>
<b>getIDs</b> .....	<b>102</b>
<b>getInheritedSoftwarePackageList</b> .....	<b>103</b>
<b>getList.</b> .....	<b>104</b>
<b>getNodeList</b> .....	<b>105</b>
<b>getParent</b> .....	<b>106</b>
<b>getPlatformList</b> .....	<b>107</b>
<b>getServerList</b> .....	<b>108</b>
<b>getSoftwarePackageIDOverrideTupleList.</b> .....	<b>109</b>
<b>getSoftwarePackageList</b> .....	<b>110</b>
<b>move</b> .....	<b>111</b>
<b>setCustomers</b> .....	<b>113</b>
<b>setPlatforms.</b> .....	<b>115</b>
<b>setSoftwarePackages.</b> .....	<b>117</b>
<b>Chapter 9: Platform Web Service</b>	<b>119</b>
<b>get.</b> .....	<b>120</b>
<b>getIDs</b> .....	<b>121</b>

<b>getList</b> .....	<b>122</b>
<b>Chapter 10: Scheduling Web Service</b>	<b>123</b>
<b>cancel</b> .....	<b>124</b>
<b>schedule</b> .....	<b>125</b>
<b>update</b> .....	<b>128</b>
<b>Chapter 11: Security Web Service</b>	<b>131</b>
<b>assignAdministratorRole</b> .....	<b>133</b>
<b>assignCustomerPermissions</b> .....	<b>134</b>
<b>assignFacilityPermissions</b> .....	<b>136</b>
<b>assignUsers</b> .....	<b>138</b>
<b>createSecurityRole</b> .....	<b>140</b>
<b>createUser</b> .....	<b>142</b>
<b>getAssignedSecurityRoleNameList</b> .....	<b>144</b>
<b>getCustomerPermissionList</b> .....	<b>145</b>
<b>getFacilityPermissionList</b> .....	<b>146</b>
<b>getSecurityRoleNameList</b> .....	<b>147</b>
<b>getUserNameList</b> .....	<b>148</b>
<b>isAssignedAdministratorRole</b> .....	<b>149</b>
<b>removeSecurityRole</b> .....	<b>150</b>
<b>removeUser</b> .....	<b>151</b>
<b>unassignAdministratorRole</b> .....	<b>152</b>
<b>unassignUsers</b> .....	<b>153</b>
<b>updatePassword</b> .....	<b>154</b>
<b>Chapter 12: Server Web Service</b>	<b>155</b>
<b>associateWithCustomer</b> .....	<b>156</b>



---

<b>delete</b> .....	<b>157</b>
<b>get</b> .....	<b>158</b>
<b>getAgentStatus</b> .....	<b>159</b>
<b>getChangeLogList</b> .....	<b>160</b>
<b>getComponents</b> .....	<b>162</b>
<b>getComponentsList</b> .....	<b>163</b>
<b>getEventList</b> .....	<b>165</b>
<b>getIDs</b> .....	<b>167</b>
<b>getList</b> .....	<b>169</b>
<b>getReachableServer</b> .....	<b>171</b>
<b>reconcile</b> .....	<b>172</b>
<b>update</b> .....	<b>173</b>
<b>Chapter 13: Software Web Service</b>	<b>175</b>
<hr/>	
<b>get</b> .....	<b>176</b>
<b>getAssociatedList</b> .....	<b>177</b>
<b>getIDs</b> .....	<b>178</b>
<b>getInstalledList</b> .....	<b>180</b>
<b>getList</b> .....	<b>181</b>
<b>Chapter 14: Complex Data Types</b>	<b>183</b>
<hr/>	
<b>CommandSession</b> .....	<b>184</b>
<b>CustomAttribute</b> .....	<b>186</b>
<b>CustomAttributeFilter</b> .....	<b>187</b>
<b>Customer</b> .....	<b>188</b>
<b>CustomerFilter</b> .....	<b>189</b>
<b>DistributedScriptArgument</b> .....	<b>190</b>
<b>DistributedScriptParameterSet</b> .....	<b>192</b>

<b>DSESessionCommandResult</b> .....	<b>194</b>
<b>DSESessionProgress</b> .....	<b>195</b>
<b>DSESessionResult</b> .....	<b>196</b>
<b>DSEServerProgress</b> .....	<b>197</b>
<b>DSEServerResult</b> .....	<b>199</b>
<b>DSEServerStatus</b> .....	<b>200</b>
<b>EmailNotificationSpec</b> .....	<b>201</b>
<b>EventBase</b> .....	<b>202</b>
<b>Facility</b> .....	<b>203</b>
<b>FacilityFilter</b> .....	<b>204</b>
<b>InstalledSoftwarePackage</b> .....	<b>205</b>
<b>ListBase</b> .....	<b>206</b>
<b>Node</b> .....	<b>207</b>
<b>NodeFilter</b> .....	<b>210</b>
<b>Permission</b> .....	<b>211</b>
<b>Platform</b> .....	<b>212</b>
<b>PlatformFilter</b> .....	<b>214</b>
<b>ReconcilePackageBase</b> .....	<b>215</b>
<b>ReconcilePackageDetails</b> .....	<b>216</b>
<b>ReconcilePackageOutput</b> .....	<b>218</b>
<b>ReconcileServerParameterSet</b> .....	<b>219</b>
<b>ReconcileServerProgress</b> .....	<b>221</b>
<b>ReconcileServerResult</b> .....	<b>222</b>
<b>ReconcileServerResultBase</b> .....	<b>224</b>
<b>ReconcileServerResultPreview</b> .....	<b>226</b>
<b>ReconcileSessionArgument</b> .....	<b>227</b>
<b>ReconcileSessionProgress</b> .....	<b>228</b>

---

<b>ReconcileSessionResult</b> .....	<b>229</b>
<b>ReconcileSessionResultBase</b> .....	<b>230</b>
<b>ReconcileSessionResultPreview</b> .....	<b>231</b>
<b>Script</b> .....	<b>232</b>
<b>ScriptVersion</b> .....	<b>234</b>
<b>ScheduleSpec</b> .....	<b>235</b>
<b>Server</b> .....	<b>237</b>
<b>ServerChangeLog</b> .....	<b>242</b>
<b>ServerComponents</b> .....	<b>243</b>
<b>ServerEvent</b> .....	<b>246</b>
<b>ServerFilter</b> .....	<b>247</b>
<b>SoftwarePackage</b> .....	<b>249</b>
<b>SoftwarePackageFilter</b> .....	<b>252</b>
<b>SoftwarePackageIDOverrideTuple</b> .....	<b>253</b>
<b>Appendix A: Compatibility With Previous Releases</b>	<b>255</b>
<b>Running API Clients Developed for Previous Releases</b> .....	<b>255</b>
<b>Authentication and Authorization in Releases Prior to 2.1</b> .....	<b>255</b>
<b>Exception Handling in Releases Prior to 2.1</b> .....	<b>256</b>
<b>Changes in Operation Names From Version 1.0 to 2.0</b> .....	<b>256</b>



# Preface

Document Date: 8/22/05

Welcome to the Opsware SAS Web Services API 2.2 – a standards-based programming environment built on open industry standards such as SOAP (Simple Object Access Protocol) and WSDL (Web Services Definition Language).

The Opsware SAS Web Services API 2.2 enables the integration of applications and other systems with the Opsware Server Automation System (SAS). This broadens the scope of how IT can use Opsware SAS to achieve operational goals.

## About this Guide

This guide describes how to use the Opsware SAS Web Services API 2.2, starting with an overview of the API. It provides information about using the API to manage servers, reconcile servers (install or remove software), manipulate the software node tree, and execute automated scripts.

## Audience for this Guide

This guide is intended for software developers who will be writing programs using the Opsware SAS Web Services API 2.2.

To use the Opsware SAS Web Services API 2.2, you should:

- Be familiar with Opsware SAS (have taken the Opsware SAS Fundamentals course or have equivalent experience).
- Have experience in programming a client application that interacts with a Web Service in one of the languages (Java, Python, Perl, or C#) that the Opsware SAS Web Services API 2.2 supports.

## Contents of this Guide

This guide contains the following chapters:

- **Chapter 1:** Overview - provides an overview description of the Opsware SAS Web Services API 2.2, including architecture diagrams and descriptions of the sample client programs
- **Chapter 2:** Set Up - discusses software requirements, security, exception handling, logging
- **Chapters 3 - 14** have reference information for each SOAP operation in a given Web Service. Each chapter begins with a brief overview of the Web Service, followed by the sections that describe the results and arguments of every operation. (See “Web Service Operations and WSDL Files” on page xv.)
- **Chapter 3:** Command Session Web Service
- **Chapter 4:** Custom Attribute Web Service
- **Chapter 5:** Customer Web Service
- **Chapter 6:** Distributed Script Web Service
- **Chapter 7:** Facility Web Service
- **Chapter 8:** Node Web Service
- **Chapter 9:** Platform Web Service
- **Chapter 10:** Scheduling Web Service
- **Chapter 11:** Security Web Service
- **Chapter 12:** Server Web Service
- **Chapter 13:** Software Web Service
- **Chapter 14:** Complex Data Types - provides reference information for the SOAP complex types used by the Opsware SAS Web Services API 2.2. (See “Complex Data Types and WSDL Files” on page xvi.)

### Conventions in this Guide





This guide uses the following typographical and formatting conventions.

NOTATION	DESCRIPTION
<b>Bold</b>	Defines terms.
<i>Italics</i>	Identifies guide titles and provides emphasis.

NOTATION	DESCRIPTION
Courier	Identifies text of displayed messages and other output from Opsware programs or tools.
Courier Bold	Identifies user-entered text (commands or information).
<i>Courier Italics</i>	Identifies variable user-entered text on the command line or within example files.

### Icons in this Guide

This guide uses the following iconographic conventions.

ICON	DESCRIPTION
	This is a note. This icon identifies especially important concepts that warrant added emphasis.
	This is a requirement. This icon identifies a task that must be performed before an action under discussion can be performed.
	This is a tip. This icon identifies information that can help simplify or clarify tasks.
	This is a warning. This icon is used to identify significant information that must be read before proceeding.

## Documentation for the Opsware SAS Web Services API 2.2

The documentation set supporting the API includes:

- This reference guide – *Opsware SAS Web Services API 2.2 Guide*
- The developer syntax docs for each client language – Java, Python, Perl, or C#

For more information, see “About the Developer Syntax Docs” on page 9.

- The Web Service Definition Language (WSDL) files – one WSDL file per Web Service (ten WSDLs total)

The WSDL file is the single most accurate standard-compliant description of the Web Services. It includes definitions of the data structures in the form of XML schemas, the objects that can be sent and retrieved, the operations exposed in the interface, and their corresponding input and output parameters. The section “How to Read this Guide” on page xiv describes how this guide documents the operations defined in the WSDL files.

- The source code for a sample client application for each supported language

The sample client applications show how to connect to the Web Service and how to invoke most of the getter operations, including those that use filter arguments for retrieving information.

- The README for the sample application for each client platform that describes how to set up and run the sample application for that platform

In addition to the supporting documentation set, Opsware Inc. provides the following resources for developers using the Opsware SAS Web Services API 2.2:

- Client stubs (See “Client Architecture” on page 6.)
- Proxy modules (See “Supported Client Languages and Technologies” on page 8.)
- A static WSDL file for each Web Service

## How to Read this Guide

The organization of this guide mirrors the underlying structure of the Opsware SAS Web Services API 2.2. The reference chapters of this guide describe every Web Service operation and most of the complex SOAP data types used by the operations.



---

## Web Service Operations and WSDL Files

In this guide, reference chapters document the operations for each Web Service and corresponding WSDL file. To find out the syntax of an operation in a particular language (Java, Python, Perl, or C#), refer to the developer syntax docs (such as the javadocs).

For example, The Server Web Service Operations Reference chapter documents the `getIDs` operation in a language-neutral manner by describing the operation result and argument as SOAP data types. The `ServerWebService.wsdl` file specifies the `getIDs` operation as follows:

```
<operation name="getIDs" >
  <input message="tns:getIDs" />
  <output message="tns:getIDsResponse" />
</operation>
. . .
<operation name="getIDs" >
<soap:operation
soapAction=""
style="rpc" />
<input>
  <soap:body use="encoded"
    namespace="http://www.opsware.com/osapi/2.0/com/opsware/ws/
    ejb/ServerWebService"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
<output>
  <soap:body use="encoded"
    namespace="http://www.opsware.com/osapi/2.0/com/opsware/ws/
    ejb/ServerWebService"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
. . .
<message name="getIDs" >
  <part name="serverFilter"
    xmlns:partns="java:com.opsware.ws.filter"
    type="partns:ServerFilter" />
</message>
<message name="getIDsResponse" >
  <part name="result"
    xmlns:partns="java:language_builtins.math"
    type="partns:ArrayOfBigDecimal" />
</message>
```

## Complex Data Types and WSDL Files

The Complex Data Types chapter describes the elements of the `ServerFilter` type in a language-neutral manner. A SOAP complex data type is made up of other data types. The `ServerFilter` complex type, for example, contains a `string` type named `machineID` and a `dateTime` type called `lastRequestTimeStamp`. Basic types, such as `string`, `decimal`, and `dateTime`, are not complex types. The developer syntax docs specify the complex types in each language (Java, Python, Perl, or C#). For example, the javadocs for the `ServerFilter` class specify the `machineID` and `lastRequestTimeStamp` elements by their corresponding getter and setter methods.

The following snippet of the `ServerWebService.wsdl` file shows some of the elements contained in the `ServerFilter` complex type:

```
<xsd:complexType name="ServerFilter">
  <xsd:sequence>
    <xsd:element name="machineID"
      maxOccurs="1"
      type="xsd:string"
      minOccurs="1"
      nillable="true">
    </xsd:element>
    <xsd:element xmlns:tp="java:language_builtins.math"
      name="serverIDs"
      maxOccurs="1"
      type="tp:ArrayOfBigDecimal"
      minOccurs="1"
      nillable="true">
    </xsd:element>
    <xsd:element name="lastRequestTimeStamp"
      maxOccurs="1"
      type="xsd:dateTime"
      minOccurs="1"
      nillable="true">
    </xsd:element>
    . . .
  </xsd:sequence>
</xsd:complexType>
```

## Array and List Complex Types

The WSDL files define arrays as `ArrayOf<type>`. For example, the `serverIDs` element is an `ArrayOfBigDecimal`, which is defined in `ServerWebService.wsdl` as:

---

```
<xsd:complexType name="ArrayOfBigDecimal">
  <xsd:complexContent>
    <xsd:restriction xmlns:soapenc="http://schemas.xmlsoap.org/
soap/encoding/"
      base="soapenc:Array">
      <xsd:attribute xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
wSDL:arrayType="xsd:decimal []"
      ref="soapenc:arrayType">
      </xsd:attribute>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

Because the WSDL files define every `ArrayOf<type>` as a `<type> []`, the Complex Data Types chapter does not include entries for the array types. Whenever you see `ArrayOf<type>` in a reference chapter, you may assume that it is a `<type> []`.

Also, the Complex Data Types chapter does not include entries for lists. A data type named `<type>List` embeds an array `<type> []` and extends `ListBase`, which contains the `timeStamp` element.



# Chapter 1: Overview

## IN THIS CHAPTER

This chapter provides the following overview information for the Opsware SAS Web Services API 2.2:

- Overview of the Opsware SAS Web Services API 2.2
- Architecture of the Opsware SAS Web Services API 2.2
- Supported Client Languages and Technologies
- About the Developer Syntax Docs

## Overview of the Opsware SAS Web Services API 2.2

The Opsware SAS Web Services API 2.2 exposes a Web Services interface to facilitate the integration of business and operations support systems with the Opsware platform. The Opsware SAS Web Services API 2.2 allows other IT systems – such as existing monitoring, trouble ticketing, billing, and virtualization technology – to exchange information with the Opsware platform.

Developers can use a Web Services-enabled development environment – such as Microsoft Visual Studio .NET, BEA WebLogic Workshop, or Perl scripts – to develop monitoring and reporting applications that invoke procedures through this interface.

The Opsware SAS Web Services API 2.2 is built on Web Services standard protocols, such as XML, SOAP (Simple Object Access Protocol), and WSDL (Web Services Definition Language), so that customers are not locked into proprietary protocols that make integration more complex and costly.

## Overview of Features

The Opsware SAS Web Services API 2.2 provides the following features:

- A secure, programmable system architecture that enables programs to interact with Opsware SAS.

- Product features written with programmability in mind to enable customers and systems integrators to extend features.
- A programmable environment that includes over ten Web Services for major Opsware SAS functionality
- Sample code and developer syntax docs for each supported client language

With the Opsware SAS Web Services API 2.2, you can:

- Build new automation applications and extend Opsware SAS applications to improve IT productivity and comply with your own IT policies and procedures. You can use the Opsware SAS Web Services API 2.2 to drive external systems (such as network or storage automation systems) as well as to be driven by external systems.
- Integrate Opsware SAS with other IT systems.
- Use the Opsware Model Repository as a central repository for storing and organizing critical IT information (operations, environment and asset information), making the repository the system of record.
- Automate a wide range of software and operating systems without having to wait for Opsware, Inc. to deliver out-of-the-box support for a particular technology.

## What's New in Version 2.2

Version 2.2 has the following new features:

- New operations in the Distributed Script Web Service:
  - `create`: Creates a new `Script` object, which represents a distributed script that can be executed on managed servers..
  - `delete`: Deletes a `Script` object.
  - `get`: Retrieves a `Script` object.
  - `getCurrentVersion`: Retrieves the current version of a script as a `ScriptVersion` object.
  - `getScriptText`: Retrieves the text (source code) of a script's current version.
  - `update`: Updates the name and usage notes of a `Script` object.
  - `updateScriptText`: Updates the text (source code) of a `Script` object, creating a new version.
- A different context URI

The URL contains the version number. In version 2.2, the API uses the following URL:

```
https://<host>:<port>/osapi/2.2/com/opsware/ws/ejb/  
<WebServiceName>?wsdl
```

## Web Services in this Version

Each Web Service in the Opsware SAS Web Services API 2.2 provides a set of operations that can be invoked through remote procedure calls. The signature of each operation includes arguments that take the form of either a unique identifier for an object or a filter that constrains the requested list of objects.

The Opsware SAS Web Services API 2.2 consists of the following Web Services:

- **Command Session Web Service:** Allows you to access progress and result information about two important Opsware Command Engine sessions: Distributed Script Execution (DSE) and reconcile. The progress information is available while the session is running, and the results are available after the session has completed execution.
- **Custom Attribute Web Service:** Used to apply special properties as name-value pairs to servers, nodes, customers, facilities, and OS definitions. You can get, set, or remove custom attributes.
- **Customer Web Service:** Enables you to create, remove, and retrieve customer accounts, the entities that allow you to create business units to provide management of Opsware SAS operations and configurations. You can also associate an Opsware customer with a facility.
- **Distributed Script Web Service:** Runs a script on one or more managed servers. The Distributed Script Execution (DSE) subsystem of Opsware supports these types of scripts: Unix/Linux shell, Windows batch (.BAT), and Windows Visual Basic (VBScript).
- **Facility Web Service:** Allows you to create, update, retrieve, or decommission an Opsware facility, which represents a collection of servers and server groups that are managed by a single Model Repository. Typically, a facility corresponds to a specific data center.
- **Node Web Service:** Used to create and delete a node in the Software Tree, the structure that organizes Opsware entities. The hierarchy of the tree enables nodes such as software applications to inherit properties from other software application nodes. An Opsware entity is an object such as `Customer`, `Facility`, `Software Package`, and `Server`. To specify relationships, you can attach entities to nodes. For example,

you can attach servers, software packages, customers, platforms to a Custom Application node.

- **Platform Web Service:** Retrieves information about a platform, which in Opsware SAS refers to an operating system.
- **Scheduling Web Service:** Schedules a previously defined function (snapshot, audit, or application configuration push) to be performed either once or repeatedly.
- **Security Web Service:** Enables you to perform security operations such as creating and removing Opsware security roles (groups) and users. You can also assign customer and facility permissions to roles.
- **Server Web Service:** Allows you to perform a variety of operations on an Opsware managed server, such as associating a server with a customer, checking the status of the Opsware Agent on the server, retrieving a list of changes for a server, and reconciling servers. (A reconcile operation installs or uninstalls software on a managed server.)
- **Software Web Service:** Retrieves information about the software packages that are defined in Opsware SAS or installed on managed servers.

## Version History

The following table lists the versions of the Opsware SAS Web Services API.

Table 1-1: Versions of the Opsware SAS Web Services API

OPSWARE SAS WEB SERVICES VERSION	OPSWARE SAS VERSION	MAJOR CHANGES
2.2 (2.1, 2.0, and 1.0 included for compatibility.)	5.2	Addition of new operations to the Distributed Script Web Service.
2.1 (2.0 and 1.0 included for compatibility.)	5.1	Addition of the Scheduling Web Service.  Opsware Exceptions.



Table 1-1: Versions of the Opsware SAS Web Services API

OPSWARE SAS WEB SERVICES VERSION	OPSWARE SAS VERSION	MAJOR CHANGES
2.0	4.6, 4.7	<p>Web Service split into multiple Web Services.</p> <p>Operation name changes. (See “Changes in Operation Names From Version 1.0 to 2.0” on page 256.)</p> <p>Versioning scheme for backward compatibility.</p> <p>Addition of reconcile, distributed script execution, and move node operations.</p>
1.0	4.5	<p>Addition of the Security Web Service.</p> <p>Minor changes to the operation names in existing Web Services.</p>
0.3, 0.2, 0.1	4.0.1, 4.0.2, 4.0.3	<p>Technology previews of the Opsware System Web Services API.</p>

For more information, see See “Compatibility With Previous Releases” on page 255.

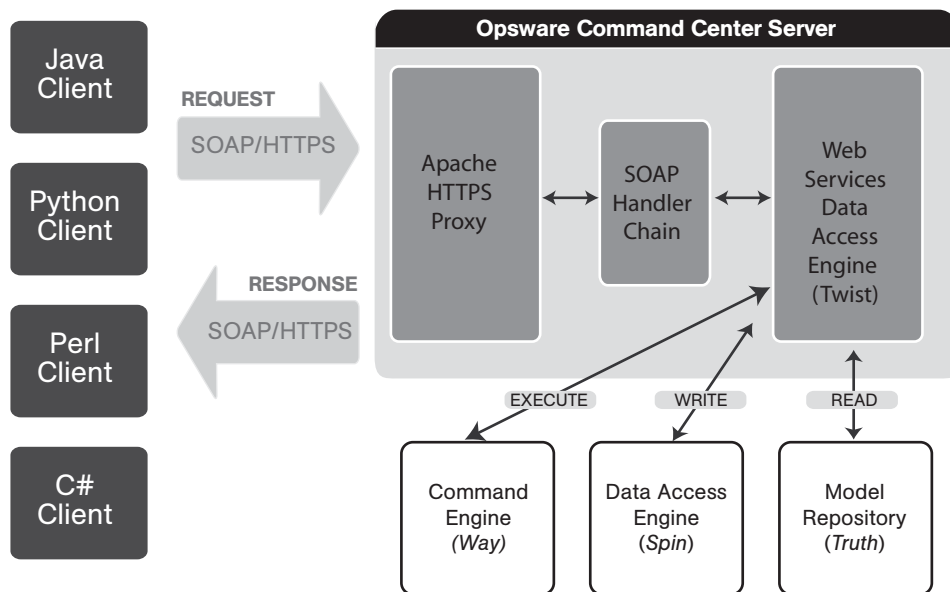
## Architecture of the Opsware SAS Web Services API 2.2

Figure 1-1 shows the overall architecture of the Opsware SAS Web Services API 2.2. The Opsware SAS Web Services API 2.2 is implemented as two sub-components. The first sub-component is a SOAP handler chain. The second sub-component is the EJB session facade layer that is implemented on the Web Services Data Access Engine (twist), which runs on WebLogic. SOAP Web Services messages are transmitted over HTTPS through

the Apache HTTPS reverse proxy. The SOAP handler chain, Web Services Data Access Engine, and Apache HTTPS reverse proxy are installed on the same core server as the Opware Command Center component.

Using SOAP over HTTPS, remote clients send requests to the Opware platform and get a response back. The SOAP handler chain receives the SOAP request message through the Apache proxy and invokes a method in the Web Services Data Access Engine. The SOAP handler chain constructs the SOAP response message and then dispatches it to the client. The response sent to the client is in the form of a SOAP envelope, the body of which contains the data operation, a report on the outcome of the request, or a SOAP fault.

Figure 1-1: Architecture of the Opware SAS Web Services API 2.2



### Client Architecture

A Web Service client consists of three components:

- Client application

A client application is written in one of these programming languages: Java, Python, Perl, or C#. A client application makes remote calls to Opware SAS by invoking the operations of the Opware SAS Web Services API 2.2. For example, a client application might invoke operations on the Server Web Service to retrieve information about the

servers managed by Opware SAS. A client application is created by a software developer, typically a system integrator, who is familiar with Opware SAS.

- Proxy module

A proxy module is a software library that implements SOAP messaging for a specific programming language. The Python ZSI proxy module, for example, converts native Python datatypes to and from SOAP messages. At runtime, the proxy module establishes the connection with the server. For a list of supported proxy modules, see Table 1-2 on page 8.

- Client stub

The client stub defines the operation signatures and data types that are available in the Opware SAS Web Services API 2.2. A client application invokes the Web Services operations on the client stub. Opware, Inc. provides client stubs for each supported proxy module.

Perl clients use dynamic proxies and do not require static stubs.

All three client-side components-- the client application, proxy module, and client stub-- must be compiled together.

## WSDL Files and Web Services URLs

The Web Services are described in the Web Services Description Language (WSDL) format. Each WSDL file provides a complete description of the operations, arguments, return values, and complex data types of a specific Web Service. (Examples of complex data types are search filters, lists, arrays, and Opware entities such as customers, facilities, and servers.) The `ServerWebService.wsdl` file, for example, defines the Opware Server Web Service.

To access the Opware SAS Web Services API 2.2, application clients specify the URL of the WSDL file for a particular service. The syntax of the WSDL file URL follows:

```
https://<occ-host>:<port>/osapi/2.2/com/opware/ws/ejb/<web-service>?wsdl
```

In the preceding URL, `<occ-host>` is the host on which the Opware Command Center core component has been installed. The `<port>` is 443, the default HTTPS port for the Apache reverse proxy that is bundled with the Opware Command Center component. The `<web-service>` is one of the following:

CommandSessionWebService  
 CustomAttributeWebService  
 CustomerWebService  
 DistributedScriptWebService  
 FacilityWebService  
 NodeWebService  
 PlatformWebService  
 SchedulingWebService  
 SecurityWebService  
 ServerWebService  
 SoftwareWebService

For example, the following URL specifies the host `occ.c07.dev.opsware.com`, the default port number, and the WSDL location for the `ServerWebService`:

```
https://occ.c07.dev.opsware.com:443/osapi/2.2/com/opsware/ws/ejb/ServerWebService?wsdl
```

## Supported Client Languages and Technologies

The Opsware SAS Web Services API 2.2 supports client applications written in Java, Python, Perl, or C#. The following table lists the specific versions of proxy modules that are supported for these client languages.

Table 1-2: Supported Proxy Modules for Client Applications

CLIENT LANGUAGE	SUPPORTED CLIENT PROXY MODULE	PROVIDER
Java	JBoss 4.0 (Axis 1.1) Web Services client	JBoss, Inc.
Python	Zolera SOAP Infrastructure (ZSI) 1.6	open source
Perl	SOAP::Lite 0.65_6	open source
C#	Microsoft .NET Framework SDK version 1.1	Microsoft Corp.



You can run the client application on any server – you are not limited to running it on a server that is managed by Opsware SAS (a server that has an Opsware Agent installed on it).

---

## About the Developer Syntax Docs

For each supported client-application language (Java, Python, Perl, or C#), Opsware Inc. provides developer docs that define the language-specific syntax of the operations in each Web Service. These developer syntax docs were generated by tools that read the client stubs provided by Opsware Inc. Although they define the syntax for the operations, the developer docs do not provide semantic (descriptive) information. For descriptions of the operation calls, parameters, and return types, see the Web Services reference chapters in this guide.

### Java Developer Docs (javadocs)

The Java developer docs are generated from the Java client stub using the `javadoc` tool. The Java docs describe:

- The interfaces of the stateless session EJBs implementing the Web Services
- The data structures for the input and output parameters of the operations in the Web Services interface

Only the public methods of the classes in the `com.opsware.ws.ejb` package are exposed as Web Service operations. These public methods have corresponding operation definitions in the WSDL files. The constructors and setters for Java beans in the other packages documented in the Java docs affect only the local objects in the client application, not the objects on the Opsware platform (server side). For example, the `Server.setAssetTag` method changes the `Server` object in the client application, but does not alter the corresponding object on the Opsware platform.

To view the methods of a specific Web Service:

- 1** In a browser window, go to the main index page for the Java docs of this release.
- 2** In the upper left pane, select `All Classes`.
- 3** In the lower left pane, select `<name>WebService`.

### Python Developer Docs

The Python developer docs are generated from the Python client stub using Epydoc, a tool for generating API documentation for Python modules. When generating the Python client stub, the ZSI proxy module creates hundreds of classes in the client stub (four classes per operation plus additional classes for the objects). Epydoc generates an HTML file for each class.

To view the methods of a specific Web Service:

- 1 In a browser window, go to the main index page for the Python docs of this release.
- 2 In the upper left pane, select the module `<name>WebService_services`.
- 3 In the lower left pane, select the class  
`<name>WebServicePortSoapBindingSOAP`.

## Perl Developer Docs

The Perl developer docs are generated from the Perl client stub using the Pdoc library.

To view the methods of a specific Web Service:

- 1 In a browser window, go to the main index page for the Perl modules docs of this release.
- 2 In the lower left pane, select `<name>WebService`.  
  
The methods (from the stub generated by SOAP::Lite) are listed in the section "Privates (from my definitions)."
- 3 Alternatively, at the top part of the page in the Toolbar section, select `Raw content`.
- 4 A page appears that displays the method signatures in a simpler format.  
  
The method parameters are included in the parameters field of each method. You can ignore the endpoint field because it is overwritten in the client application code with the appropriate endpoint location URL.

## C# Developer Docs

The C# developer docs are generated from the C# client stub using NDoc, an extensible code documentation generation tool for .NET developers. You can view the C# developer docs as MSDN-online-style web pages (generated by Opware Inc.). When generating the C# client stub, the Microsoft .NET Framework SDK creates two additional operations for each Web Service operation.

To view the methods of a specific Web Service:

- 1 In a browser window, go to the main index page for the C# docs of this release.
- 2 In the left pane, expand `<name>WebServiceProxy`.
- 3 In the left pane, expand `<name>WebService Class`.
- 4 In the left pane, select `<name>WebService Members`.
- 5 In the right pane, scroll down to the section `Public Instance Methods`.

Note each method name that starts with a lower case letter. These are the methods for the Web Service operations.

Ignore the following methods:

- Inherited methods. The C# stub contains methods inherited from classes in the .NET Framework SDK.
  - Methods that start with `Begin` or `End`. For example, a `get` method will have a `BeginGet` and an `EndGet` method generated in the stub in addition to the actual `get` method.
- 6** To view the C# signature of a method, select the method name.





# Chapter 2: Setup

## IN THIS CHAPTER

This chapter describes the following setup and usage information:

- Setup Requirements
- Authentication and Authorization
- Exception Handling
- Verbose Mode
- Logging
- About the Sample Applications

## Setup Requirements

This section discusses the prerequisites for clients of the Opsware SAS Web Services API 2.2. The README files included with the sample applications also contain setup information.

### Setup Requirements for All Client Languages

- 1** Verify that the version of the Opsware SAS Web Services API 2.2 that your clients rely on is compatible with the version of the Opsware core to be accessed by the clients. To check the version numbers, see Table 1-1, “Versions of the Opsware SAS Web Services API,” on page 4.
- 2** Make sure that the Opsware Web Services Data Access Engine is running and that the API is accessible by viewing one of the WSDL files in a browser. For an example URL, see “WSDL Files and Web Services URLs” on page 7.
- 3** Note the host and port in the URL you specified in the preceding step. Your client applications will specify these values in the WSDL URL when accessing the Opsware SAS Web Services API 2.2.

## Java Client Setup

To set up your development environment for Java clients, you need the files listed in this section. Opware Inc. provides a package (`j2ee.tar.gz`) that contains all of the required files and libraries.

### Java Client Proxy Module

- JBoss /Axis Web Services client - `axis.jar`
- Available from Opware Inc. or from the Axis Web site (open source).

### Java Client Stub

- Available from Opware Inc. - `osapi21client.jar`.

### Java Client Required Libraries

- Available from Opware Inc. - `opware_sslconfig.jar`
- Axis jar files. After you unzip the Axis download file, the following jar files are in the `lib` directory:

```
activation.jar
axis.jar
commons-discovery.jar
commons-logging.jar
jaxrpc.jar
mail.jar
ssaj.jar
wsdl4j.jar
```

### Java Client Security Requirements

Java clients require a trusted certificate to verify the server certificate presented to the client during the SSL handshake. Copy the Opware trusted server certificate file, `opware-ca.crt`, from the Opware Web Services Data Access Engine server directory to the client home directory. The Opware Web Services Data Access Engine server directory is as follows:

```
/var/1c/crypto/twist
```

Make sure that the `CLASSPATH` used by the Java client includes the client home directory.

## Python Client Setup

To set up your development environment for Python clients, you need the files listed in this section. Opsware Inc. provides a package (`python.tar.gz`) that contains the required client stub and types files. Each Web Service has a stub file and a types file.

### Python Software Dependencies

Install the following software for your development environment:

- Python 2.4
- `fpconst-0.6.0`
- `PyXML-0.8.4`

Follow the installation instructions in the README files of these packages.

### Python Client Proxy Module

- ZSI (Zolera SOAP Infrastructure) - open source
- Current version: 1.6

### Python Client Stub

You must use the stub provided by Opsware Inc. This ZSI-based stub was generated and tested with Python 2.4. (The `wsdl2py` stub generator script does not work properly with Python 2.2.) Opsware Inc. patched this stub because the float format for decimal caused certain operations to fail in the Opsware Web Services Data Access Engine.

ZSI generates the stub and types files from a WSDL file. The types file contains type definitions for the data types described in the WSDL. Opsware Inc. provides a stub file and a types files for each service. For example, the provided files for the Customer Web Service are as follows:

```
CustomerWebService_services.py (stub file)
CustomerWebService_services_types.py (types file)
```

### Additional Python Client Requirements

- Set the `PYTHONPATH` environment variable to include the location path names for `fbconst-0.7.0`, `PyXML-0.8.4`, and `ZSI-1.6.0`. For example, in `cs` you might set `PYTHONPATH` as follows:

```
setenv PYTHONPATH /h/foo/wsapi/zsi/fpconst-0.6.0:/h/foo/wsapi/soappy/PyXML-0.8.4:/h/foo/wsapi/zsi/ZSI-1.6.0
```

- Using the SOAPpy proxy module to generate the Python client stub is not supported. The SOAPpy proxy module does not correctly serialize complex objects, including the search filters of the Opware SAS Web Services API 2.2.

### **Perl Client Setup**

To set up your development environment for Perl clients, you will need the files listed in this section.

A Perl client does not require a static client stub to access the Opware SAS Web Services API 2.2. The Perl proxy module uses a dynamic proxy to connect to the server. Therefore, you do not need to obtain Perl stub files from Opware Inc.

### **Software Dependencies**

Install the following software for your development environment:

- Crypt-SSLeay-0.51
- IO-Socket-SSL-0.95
- Net\_SSLeay.pm-1.25
- HTML-Parser-3.35
- MIME-Base64-3.01
- URI-1.30
- libwww-perl-5.76

Follow the installation instructions in the README files of these packages.

### **Perl Client Proxy Module**

- SOAP::Lite for Perl - open source
- Current version: SOAP-Lite-0.65\_6

### **Perl Client Stub**

- A static stub is not required.

### **Known Issues**

- User credentials must be encoded in the WSDL URL for Basic Authentication.
- Other mechanisms for specifying user credentials do not work.

## **C# Client Setup**

To set up your development environment for C# clients, you must meet the following requirements. Opware Inc. provides a package (`osapi21sample-csharp.zip`) that contains the required client stub file, `App.config` file, and certificate validation class. For instructions on setting up and running the sample client, see the README file included in `osapi21sample-csharp.zip`.

## **C# Software Dependencies**

Install the following software for your development environment:

- Microsoft .NET Framework SDK version 1.1

## **C# Client Stub**

- Opware Inc. provides a stub file for each service, for example:  
`CustomerWebService.cs`

## **C# Client Proxy Module**

- Microsoft .NET Framework SDK version 1.1.

## **Authentication and Authorization**

Users of the Opware SAS Web Services API 2.2 must be authenticated and authorized to invoke operations on the API. For authentication and authorization, the API relies on the same security mechanism and domain as the Opware Command Center. You should use the Opware Command Center to specify the users and permissions required by the API clients.

If a client submits an incorrect user/password combination, authentication fails and an `OpwareAuthenticationException` is thrown. If the user does not have permission to perform an operation, authorization fails and an `OpwareAccessControlException` is thrown.

For more information, see “Authentication and Authorization in Releases Prior to 2.1” on page 255.

## HTTPS Access

Communication between client applications and the server component of the Opsware SAS Web Services API 2.2 is encrypted. The request and response SOAP messages (which implement the Web Service operation calls) are encrypted using SSL over HTTP (HTTPS).

## Exception Handling

In the Opsware SAS Web Services API 2.2, operations may throw an `OpswareException` or one of its subclasses. These exceptions are specified in the `wSDL:fault` element within the definition of the operation. For example, in the Scheduling Web Service WSDL, the definition of the `update` operation includes the following elements:

```
<wSDL:fault message="impl:OpswareNullArgumentException"
name="OpswareNullArgumentException"/>
  <wSDL:fault message="impl:OpswareInternalErrorException"
name="OpswareInternalErrorException"/>
  <wSDL:fault message="impl:OpswareAppConfigException"
name="OpswareAppConfigException"/>
  <wSDL:fault message="impl:OpswareAuthenticationException"
name="OpswareAuthenticationException"/>
  <wSDL:fault message="impl:OpswareAccessControlException"
name="OpswareAccessControlException"/>
  <wSDL:fault message="impl:OpswareException"
name="OpswareException"/>
```

## SOAP Fault Elements

SOAP messages carry error information within the SOAP Fault element in the body of the SOAP message. The SOAP Fault element has these sub-elements:

- `faultcode`
- `faultstring`
- `faultactor`

Clients map the SOAP Fault element into data structures specific for each programming language. For the mapping in each language, see the tables in the sections that follow.

The Opsware SAS Web Services API 2.2 reports on runtime exceptions with an error message in the `faultstring` sub-element. On the server running the Web Services Data Access Engine, the following properties file contains the error and logging message strings:

```
/cust/twist/i18n/WebServicesAPIMessagesBundles.properties
```

### Java Client Exceptions

In the Opsware SAS Web Services API 2.2, Java clients can catch `OpswareException` or one of its subclasses, such as `OpswareNullArgumentException`. These exceptions are in the `com.opsware.ws.exception` package. Because `OpswareException` is a subclass of `AxisFault`, it inherits the fields of `AxisFault`, which correspond to the SOAP Fault sub-elements listed in the preceding section.

The following table shows the mapping of the SOAP Fault sub-elements to Java.

Table 2-3: Java Mapping for SOAP Faults

SOAP FAULT SUB-ELEMENT	JAVA MAPPING
<code>fautcode</code>	<code>OpswareException.getFaultCode()</code>
<code>faultstring</code>	<code>OpswareException.getFaultString()</code>
<code>faultfactor</code>	<code>OpswareException.getFaultActor()</code>

The following code snippet shows how a Java client would catch `OpswareException`:

```
try {
    Server server = serverWSEndpoint.get( serverID );
} catch ( OpswareException e ) {
    System.out.println( "Caught OpswareException." );
    System.out.println( " Fault Code: " + e.getFaultCode() );
    System.out.println( " Fault String: " + e.getFaultString()
);
    System.out.println( " Fault Actor: " + e.getFaultActor() );
}
```

## Python Client Exceptions

The following table shows the mapping of the SOAP Fault sub-elements to Python.

Table 2-4: Python Mapping for SOAP Faults

SOAP FAULT SUB-ELEMENT	PYTHON MAPPING
fautcode	code
faultstring	string
faultfactor	actor

For more information about this mapping, see the following URL:

<http://pywebsvcs.sourceforge.net/zsi.html#SECTION00900000000000000000>

The following code snippet shows how a Python client would extract the SOAP Fault sub-elements:

```
try:
    response = wsProxy.method( request )
    result = response._result
except Exception, e:
    print " Fault string: %s" % e.fault.string
    print " Fault code: %s" % e.fault.code
    print " Fault actor: %s" % e.fault.actor
```

## Perl Client Exceptions

The following table shows the mapping of the SOAP Fault sub-elements to Perl.

Table 2-5: Perl Mapping for SOAP Faults

SOAP FAULT SUB-ELEMENT	PERL MAPPING
fautcode	faultcode
faultstring	faultstring
faultfactor	faultactor

For more information about this mapping, see the following URL:

[http://cpan.uwinnipeg.ca/htdocs/SOAP-Lite/SOAP/Lite.html#soap\\_\\_fault](http://cpan.uwinnipeg.ca/htdocs/SOAP-Lite/SOAP/Lite.html#soap__fault)



The following code snippet shows how a Perl client would extract the SOAP Fault sub-elements:

```
my $soap = SOAP::Lite .....;
my $som = $soap->method(@parameters);

if ($som->fault) { # will be defined if Fault element
                  # is in the message
    $som->faultcode; #
    $som->faultstring; # also available
    $som->faultfactor; #
    $som->result; # gives you access to result of call
                  # it could be any data structure
}
```

### C# Client Exceptions

The following table shows the mapping of the SOAP Fault sub-elements to C#.

Table 2-6: C# Mapping for SOAP Faults

SOAP FAULT SUB-ELEMENT	C# MAPPING
fautcode	Code
faultstring	Message
faultfactor	Actor

For more information about this mapping, see the following URL:

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfssystemwebservicesprotocolssoapexceptionclasstopic.asp?frame=true>

The following code snippet shows how a C# client would extract the SOAP Fault sub-elements:

```
catch (SoapException error) {
    // The exception details.
    Console.WriteLine("Fault Code Namespace" +
        error.Code.Namespace);
    Console.WriteLine("Fault Code Name" + error.Code.Name);
    Console.WriteLine("SOAP Actor that threw Exception" +
        error.Actor);
    Console.WriteLine("Error Message" + error.Message);
}
```

## Verbose Mode

Perl and Python clients can invoke Web Services in verbose mode to view the SOAP envelopes as they are transmitted in the request and response messages. The following sections show how to change the sample client applications to invoke the Web Services in verbose mode.

### Perl Clients in Verbose Mode

In the following code snippet from the sample client (`wsapi.pl`), uncomment the last line and add a comma after the closing curly bracket on the previous line:

```
use SOAP::Lite +autodispatch =>
    uri => 'http://www.opsware.com/opsw/OpwareWebService',
    proxy => 'https://
integration:integration@occ.c07.dev.opsware.com:443/opsw/ws',
    on_fault => sub { my ($soap, $res) = @_;
        die ref $res ? $res->faultdetail : $soap->transport-
>status, "\n";
    }
    # trace => 'debug' # uncomment to trace SOAP messages
```

### Python Clients in Verbose Mode

In the following code snippet from the sample client (`wsapi.py`), uncomment the last line and replace the closing curly bracket on the previous line (containing `url`) with a comma:

```
kw = {'auth': (AUTH.httpbasic, username, password),
      'host': host,
      'port': port,
      'ssl': 1,
      'url': url}
      # 'tracefile': sys.stdout} # uncomment to trace SOAP
messages
```

## Logging

The Opware SAS Web Services API 2.2 writes entries for completed transactions in the two Web Services Data Access Engine log files:

- `/var/lc/twist/twist.log` – WebLogic-specific error or informational messages.
- `var/lc/twist/stdout.log` – Debug output and logging of every exception that is generated by the server. This log file does not conform to a specific format.

The log file entries from the Opsware SAS Web Services API 2.2 have the following prefix:

```
<timestamp><WebService21><webservice-name.operation-name>
```

## About the Sample Applications

For each supported client language (Java, Python, Perl, or C#), Opsware Inc. provides sample applications that demonstrate the Opsware SAS Web Services API 2.2. All the samples connect to the same Web Services and invoke the same operations. Generally, the samples illustrate how to use get operations, including the use of search filters.

For detailed information about how to modify and run the samples in your environment, see the README files included in the sample application packages. See also, “Setup Requirements” on page 13.



Your client application should be compiled with the client stub, which is provided by Opsware Inc. The client stub that you obtain from Opsware Inc. should not be modified.

## Sample Application Packages

The following table lists the contents of the sample application packages provided by Opsware Inc.

Table 2-7: Sample Application Packages

CLIENT LANGUAGE	PACKAGE FILE NAME	CONTENTS OF PACKAGE
Java	osapi21sample-java.tar.gz	WebServicesDemo.java Makefile Opsware security certificate wsapi.sh (script to run client) client stubs README
Python	osapi21sample-python.tar.gz	wsapi.py client stubs README
Perl	N/A	wsapi.pl

Table 2-7: Sample Application Packages

CLIENT LANGUAGE	PACKAGE FILE NAME	CONTENTS OF PACKAGE
C#	osapi21sample-csharp.zip	WebServicesDemo.cs MyCertificateValidation.cs App.config client stubs README

### Steps Performed by the Sample Client Applications

Each sample client application performs the following steps:

- 1 Initializes an SSL client.

All clients must use SSL to connect to the Opsware SAS Web Services API 2.2. Java clients must validate the trusted server certificate of Opsware SAS by setting a JSSE adapter to use the SSL client.

- 2 Binds to the Server and Custom Attribute Web Services by providing the URLs of the services.

In the sample clients, the URLs are constructed by concatenating string values that are either hard coded in the source code, set in a shell script, or specified in an application configuration file (C#). You must modify the URL's host name and you may need to modify the port number. For instructions on modifying the URLs, see the README files included with the sample clients. See also, "WSDL Files and Web Services URLs" on page 7. foo

The client application binds to a Web Service only once, and uses the port binding to invoke operations for that Web Service

- 3 Invokes the following Server Web Service operations and prints the results:

```
get
getAgentStatus
getComponents
getIDs
getList
```

The client passes the `serverFilter` argument to the `getIDs` and `getList` operations.

For more information about these operations, see “Server Web Service” on page 155. For more information about the `serverFilter`, see “Complex Data Types” on page 183.

- 4** Invokes the following Custom Attribute Web Service operations and prints the results:

```
set  
get  
getKeyList  
getList  
remove
```

The client passes the `customAttributeFilter` argument to the `set` and `remove` operations.

For more information about these operations, see “Custom Attribute Web Service” on page 43. For more information about the `customAttributeFilter`, see “Complex Data Types” on page 183.



## Chapter 3: Command Session Web Service

### IN THIS CHAPTER

This chapter describes the following SOAP operations:

- get
- getDSESessionCommandResult
- getDSEServerProgress
- getDSEServerResult
- getDSESessionProgress
- getDSESessionResult
- getReconcileServerProgress
- getReconcileServerResult
- getReconcileServerResultPreview
- getReconcileSessionProgress
- getReconcileSessionResult
- getReconcileSessionResultPreview
- getStatus

The Command Session Web Service allows you to access progress and result information about two important Command Engine sessions: Distributed Script Execution (DSE) and reconcile. The progress information is available while the session is running. The results are available after the session has finished running.

The Opsware Command Engine runs distributed programs across many servers by interacting with Opsware Agents installed on the servers. Command Engine scripts are written in Python and run on the Command Engine server.

DSE provides the ability to configure scripts to run simultaneously across multiple Unix or Windows servers, and monitor script execution on each server. After a script runs, job- and server-specific execution results are available for review. To run a script, invoke the `DistributedScript.execute` operation.

The `reconcile` function updated the actual software configuration of a server based on the specified configuration stored in the Model Repository. During a reconcile, software, applications, and patches may be installed or uninstalled on one or more servers. To launch a reconcile, invoke the `Server.reconcile` operation.



## get

Gets the specified `CommandSession` object.

The results of a Command Engine session provide such information as the name of the agent through which the session was run, the start and end date of the session, the person who initiated the session, session descriptions, the results digest, and so on.

## Result

SOAP Data Type: `CommandSession`

Description: A `CommandSession` object, which contains information about a specific Command Engine session.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	The ID of the <code>CommandSession</code> object to retrieve.

## Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `sessionID` is null.

`OpswareObjectNotFoundException` - is thrown if the `CommandSession` is not found.

## getDSESessionCommandResult

Gets the command results (stdout, stderr) of a distributed script execution (DSE).

### Result

SOAP Data Type: `DSESessionCommandResult`

Description: Results information about a DSE session command

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
commandID	decimal	ID of the command.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `commandID` is null.

`OpswareObjectNotFoundException` - is thrown if the `DSESessionCommandResult` is not found.

## getDSEServerProgress

Gets progress information of a DSE session for an individual server, including the number of completed steps, error details, script status, and so on.

### Result

SOAP Data Type: DSEServerProgress

Description: Progress information about a current DSE session for an individual server.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
sessionID	decimal	ID of the DSE session from which to retrieve server progress information.
serverID	decimal	ID of the server from which to retrieve DSE session progress information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`OpawareNullArgumentException` - is thrown if `sessionID` or `serverID` is null.

`OpawareObjectNotFoundException` - is thrown if the `DSEServerProgress` is not found.

## getDSEServerResult

Gets the results of a DSE session for a specified server, including the script execution, errors (if there are any) and status).

### Result

SOAP Data Type: DSEServerResult

Description: Results information about a DSE session for an individual server.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
sessionID	decimal	ID of the DSE session results to retrieve for an individual server.
serverID	decimal	ID of the server from which to retrieve DSE session results.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`OpwareNullArgumentException` - is thrown if `sessionID` or `serverID` is null.

`OpwareObjectNotFoundException` - is thrown if the `DSEServerResult` is not found.

## getDSESessionProgress

Gets the progress status of a current DSE session, including the total scripts run for each server as the scripts executes.

### Result

SOAP Data Type: DSESessionProgress

Description: Progress information about a DSE session.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
sessionID	decimal	ID of the DSE session from which to retrieve progress information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`OpawareNullArgumentException` - is thrown if `sessionID` is null.

`OpawareObjectNotFoundException` - is thrown if the `DSESessionProgress` is not found.

## getDSESessionResult

Gets the results of a DSE session, such the status and results of each server affected by the session, any errors that were encountered, and so on.

### Result

SOAP Data Type: `DSESessionResult`

Description: Results information about a DSE session.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	ID of the DSE session from which to retrieve results.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `sessionID` is null.

`OpswareObjectNotFoundException` - is thrown if the `DSESessionResult` is not found.

## getReconcileServerProgress

Returns progress status of a reconcile session for an individual server.

### Result

SOAP Data Type: `ReconcileServerProgress`

Description: Progress information for a current reconcile session on a server.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	ID of the reconcile session from which to retrieve progress information.
<code>serverID</code>	<code>decimal</code>	ID of the server from which to retrieve reconcile progress information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `sessionID` or `serverID` is null.

`OpswareObjectNotFoundException` - is thrown if the `ReconcileServerProgress` is not found.

## getReconcileServerResult

Gets the results information of a reconcile session for an individual server, such as details about the packages installed or uninstalled on the server.

### Result

SOAP Data Type: ReconcileServerResult

Description: Results information about script execution for an individual server.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
sessionID	decimal	ID of the reconcile session from which to get results for a specified server.
serverID	decimal	ID of the server from which to get reconcile results information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `sessionID` or `serverID` is null.

`OpswareObjectNotFoundException` - is thrown if the `ReconcileServerResult` is not found.



## getReconcileServerResultPreview

Gets the results of a reconcile preview session for an individual server. The preview reconcile, which is run before each full reconcile, allows you to see exactly what will happen to the server as a result of the software requested to be installed or uninstalled. A preview shows what will occur if the reconcile is initiated, what software will be installed and uninstalled, details about the software packages, and so on.

### Result

SOAP Data Type: `ReconcileServerResultPreview`

Description: Preview information about a reconcile session.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	ID of the reconcile preview session to retrieve.
<code>serverID</code>	<code>decimal</code>	ID of the server from which to retrieve reconcile preview information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`OpawareNullArgumentException` - is thrown if `sessionID` or `serverID` is null.

`OpawareObjectNotFoundException` - is thrown if the `ReconcileServerResultPreview` is not found.

## getReconcileSessionProgress

Gets the progress status of the reconcile session, including how many servers have been reconciled and progress information for each server affected by the session.

### Result

SOAP Data Type: ReconcileSessionProgress

Description: Details about the progress of a current reconcile session.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
sessionID	decimal	ID of the reconcile session from which to retrieve progress information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `sessionID` is null.

`OpswareObjectNotFoundException` - is thrown if the `ReconcileSessionProgress` is not found.

## getReconcileSessionResult

Gets the results of a reconcile session, including how many servers successfully completed the reconcile, details about the software packages installed, and so on.

### Result

SOAP Data Type: `ReconcileSessionResult`

Description: Result information from a reconcile session.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	ID of the reconcile session to retrieve results of.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`OpwareNullArgumentException` - is thrown if `sessionID` is null.

`OpwareObjectNotFoundException` - is thrown if the `ReconcileSessionResult` is not found.

## getReconcileSessionResultPreview

Gets the results of a reconcile session preview. The preview reconcile, which is run before each full reconcile, allows you to see exactly what will happen to the server as a result of the software requested to be installed or uninstalled.

Preview reconcile shows what packages will be installed and what packages will be removed. If a package is removed or installed as a result of another package being installed, this information is noted as a “side effect.”

Preview reconcile also indicates which package installs or uninstalls require a reboot.

### Result

SOAP Data Type: `ReconcileServerResultPreview`

Description: Reconcile preview results information.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	ID of the reconcile preview session results to retrieve.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`OpwareNullArgumentException` - is thrown if `sessionID` is null.

`OpwareObjectNotFoundException` - is thrown if the `ReconcileServerResultPreview` is not found.

## getStatus

Gets the status of the specified `CommandSession` object.

### Result

SOAP Data Type: `string`

Description: Text message indicating the status of the `CommandSession` object.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>sessionID</code>	<code>decimal</code>	The ID of the <code>CommandSession</code> object from which to retrieve status information.

### Exceptions

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`OpswareNullArgumentException` - is thrown if `sessionID` is null.

`OpswareObjectNotFoundException` - is thrown if the `CommandSession` is not found.



# Chapter 4: Custom Attribute Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- get
- getKeyList
- getList
- getValues
- remove
- set

A custom attribute is a name-value pair used to apply special properties to servers, nodes, customers, facilities, and OS definitions. (The name of a custom attribute is also referred to as the key.) Custom attributes provide parameters to Opsware SAS for functions such as customizing displays, or providing settings during the installation or configuration of software packages.

## get

Returns a custom attribute specified by the object type, object ID, and key. All three arguments are required.

### Result

SOAP Data Type: CustomAttribute

Description: The object that represents the custom attribute.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
objectType	string	The type of Opsware entity the custom attribute has been assigned to. The objectType may be one of the following: CUSTOMER FACILITY NODE SERVER
objectID	decimal	The unique ID of the Opsware entity the custom attribute has been assigned to. For example, if the objectType is CUSTOMER, then the objectID is the same as the customerID.
key	string	The name of the custom attribute to be returned.

### Exceptions

OpswareNullArgumentException - is thrown if the specified objectType, objectID, or key is null.



`OpawareInvalidArgumentException` - is thrown if the specified `objectType` is invalid.

`OpawareObjectNotFoundException` - is thrown if no `CustomAttribute` object matches the specified arguments.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getKeyList

Returns custom attribute names (keys) by object type and object ID.

### Result

SOAP Data Type: CustomAttributeKeyList

Description: An array of custom attribute object names.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
objectType	string	The type of Opsware entity the custom attribute has been assigned to. The objectType may be one of the following: CUSTOMER FACILITY NODE SERVER
objectID	decimal	The unique ID of the Opsware entity the custom attribute has been assigned to. For example, if the objectType is CUSTOMER, then the objectID is the same as the customerID.

### Exceptions

OpswareNullArgumentException - is thrown if the specified objectType or objectID is null.

OpswareInvalidArgumentException - is thrown if the specified objectType is invalid.

`OpawareObjectNotFoundException` - is thrown if no `CustomAttribute` object matches the specified arguments.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered

## getList

Returns custom attributes by object type and object ID.

### Result

SOAP Data Type: CustomAttributeList

Description: An array of custom attribute objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
objectType	string	The type of Opsware entity the custom attribute has been assigned to. The objectType may be one of the following: CUSTOMER FACILITY NODE SERVER
objectID	decimal	The unique ID of the Opsware entity the custom attribute has been assigned to. For example, if the objectType is CUSTOMER, then the objectID is the same as the customerID.

### Exceptions

OpswareNullArgumentException - is thrown if the specified objectType or objectID is null.

OpswareInvalidArgumentException - is thrown if the specified objectType is invalid.

`OpawareObjectNotFoundException` - is thrown if no `CustomAttribute` object matches the specified arguments.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getValues

Returns the custom attribute values that match the criteria specified by the `customAttributeFilter` argument. All three elements in the `customAttributeFilter` must be specified.

### Result

SOAP Data Type: `CustomAttributeList`

Description: An array of custom attribute object values.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customAttributeFilter</code>	<code>CustomAttributeFilter</code>	The object that specifies the search criteria for this operation.

### Exceptions

`OpswareNullArgumentException` - is thrown if `customAttributeFilter` is null.

`OpswareInvalidArgumentException` - is thrown if the specified `objectType` field is invalid, or if the list of object IDs field is null or empty, or if the list of custom attribute keys field is null or empty.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## remove

Deletes custom attributes. The criteria to delete custom attributes are specified by the `customAttributeFilter` argument. All three elements in the `customAttributeFilter` are required.

### Result

SOAP Data Type: `string`

Description: The `string` OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customAttributeFilter</code>	<code>CustomAttributeFilter</code>	The object that specifies the search criteria for this operation

### Exceptions

`OpswareNullArgumentException` - is thrown if `customAttributeFilter` is null.

`OpswareInvalidArgumentException` - is thrown if the specified `objectType` field is invalid, or if the list of `object IDs` field is null or empty, or if the list of `custom attribute keys` field is null or empty.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## set

Sets the value of a custom attribute. All three elements in the `customAttributeFilter` argument are required. The `customAttributeFilter.customAttributeKeys` must contain a single array element. (In other words, just one key may be specified.)

## Result

SOAP Data Type: `string`

Description: The string OK.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customAttributeFilter</code>	<code>CustomAttributeFilter</code>	The object that specifies the search criteria for this operation.
<code>customAttributeValue</code>	<code>String</code>	The value to which the custom attribute will be set.

## Exceptions

`OpswareNullArgumentException` - is thrown if `customAttributeFilter` is null.

`OpswareInvalidArgumentException` - is thrown if the specified `objectType` field is invalid, or if the list of object IDs field is null or empty, or if the list of custom attribute keys field is null or empty or contains more than one key.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



# Chapter 5: Customer Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- associateWithFacility
- create
- delete
- get
- getIDs
- getList
- update

In Opware SAS, customer accounts allows to create business units to provide management of Opware SAS operations and configurations.

By default, Opware SAS is shipped with the following two customers:

- Customer Independent - A global customer in Opware SAS. Resources (applications, patches, and templates) that are associated with Customer Independent can be installed on any managed server, no matter what customer the server is associated with.
- Not Assigned - The servers are not associated with a customer. You can install applications, patches, or templates that are Customer Independent on Not Assigned servers. However, you cannot install or use any resources associated with a customer on a server that is not assigned to a customer.

## associateWithFacility

Associates a customer with a facility (causing the servers managed at that facility to become associated with the customer).

### Result

SOAP Data Type: `string`

Description: The result of this operation.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerID</code>	<code>decimal</code>	The ID of the customer to be associated with the facility.
<code>facilityID</code>	<code>decimal</code>	The ID of the facility to which the customer will be associated.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID` or `customerID` is null.

`OpswareErrorException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

**create**

Creates a new customer. After you create a customer, you can define the custom attributes for that customer.

**Result**

SOAP Data Type: decimal

Description: The ID (primary key) of the new customer.

**Arguments**

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
customerName	string	The name of the new customer in Opsware SAS. The value must be in uppercase, less than 25 characters, and cannot contain spaces.
businessCustomerID	decimal	The business ID of the new customer.
displayName	string	The display name of the new customer in the Opsware Command Center. The value must contain less than 50 characters and cannot include any special characters.
authorizationDomain	string	The authorization domain of the new customer. The value must be uppercase, less than 50 characters, and in the domain name format. This value is usually the same as the domain name.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `customerName` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## delete

Deletes a customer. Deleting a customer removes the customer information from the Model Repository, and moves deactivated servers assigned to the customer to the Not Assigned customer or deletes the data about the servers from the Model Repository.

You can only delete a customer from the Opsware Command Center when the following conditions are true:

- No Nodes are attached to the customer.
- The customer does not own any software packages; the packages uploaded for the customer must be deleted or deprecated.
- All servers assigned to the customer are deactivated
- No IP range groups are created for the customer
- No IP ranges are created for the customer
- No server groups are created for the customer.

## Result

SOAP Data Type: `string`

Description: The string OK.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTIONS
customerID	decimal	The ID of the customer to be deleted.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `customerID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

**get**

Returns a customer specified by the ID.

**Result**

SOAP Data Type: Customer

Description: The object that represents the customer.

**Arguments**

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTIONS
customerID	decimal	The ID of the customer to be returned.

**Exceptions**

`OpswareNullArgumentException` - is thrown if the specified `customerID` is null.

`OpswareObjectNotFoundException` - is thrown if no customer object matches the specified `customerID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getIDs

Returns an array of customer IDs that matches the criteria specified by the `customerFilter` argument. To get all active customers, specify a null `customerFilter`. To get specific active customers, specify the `customerFilter.customerIDs`. To search for a customer by name, specify the `customerFilter.customerName`. To search for all customers associated with a facility, specify the `customerFilter.facilityID`.

This operation searches for customers with the following algorithm:

```

if customerFilter is null
return an array with the IDs of all active customers
else if customerFilter.customerIDs is not empty
return an array with the IDs of the specified active
customers
else if customer.customerName is not null
return a single-element array with the ID of the
corresponding customer
else if customer.facilityID is not null
return an array with the IDs of all customers associated with
the specified facility
else
return an array with the IDs of all customers

```

## Result

SOAP Data Type: `decimal []`

Description: An array of customer IDs.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerFilter</code>	<code>CustomerFilter</code>	The object that specifies the search criteria for this operation.

## Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.



`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getList

Returns a list of customers that matches the criteria specified by the `customerFilter` argument.

The search performed by this operation follows the same criteria specified by the `customerFilter` argument documented for `getIDs`.

### Result

SOAP Data Type: `CustomerList`

Description: An object that represents multiple customers.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerFilter</code>	<code>CustomerFilter</code>	The object that specifies the search criteria for this operation.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## update

Edits a customer.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerID</code>	<code>decimal</code>	The ID of the customer to be edited. The ID itself cannot be changed.
<code>customerName</code>	<code>string</code>	The name of the customer to be edited in Opware SAS. The value must be in uppercase, less than 25 characters, and cannot contain spaces.
<code>businessCustomerID</code>	<code>decimal</code>	The business ID of the customer to be edited.
<code>displayName</code>	<code>string</code>	The display name of the customer to be edited. This name appears in the Opware Command Center.
<code>status</code>	<code>string</code>	The status of the customer to be edited.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `customerID` or `customerName` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## Chapter 6: Distributed Script Web Service

### IN THIS CHAPTER

This chapter describes the syntax of the following SOAP operations:

- create
- delete
- execute
- get
- getCurrentVersion
- getScriptText
- update
- updateScriptText

The Distributed Script Execution (DSE) feature enables you to run a script on one or more managed servers. DSE supports these types of scripts: Unix/Linux shell, Windows batch (.BAT), and Windows Visual Basic (VBScript).

The Opsware Command Engine handles the entry of scripts into the Opsware Model Repository and the versioning of stored scripts. During script execution on the servers, the Command Engine runs a session that issues an execution command to the Opsware Agent on each server. Each Agent handles script execution and sends results to the Command Engine, which enters the results data into the Model Repository.

For information on required permissions and other tips, see the the Distributed Script Execution chapter of the *Opsware<sup>®</sup> SAS 5.2 User's Guide*.

## **create**

Creates a new `Script` object, which represents a distributed script that can be executed on managed servers.

This operation saves the `Script` object in the Opware Model Repository. To run the script, invoke the `execute` operation.

## **Result**

SOAP Data Type: `decimal`

Description: The ID of the newly created script. When running the script with the `execute` operation, specify this ID in the `dseArguments.parameter.scriptID` argument.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptName</code>	string	The display name of the script. The <code>scriptName</code> must be unique and less than 100 characters in length.
<code>scriptText</code>	string	The text (source code) of the script.
<code>codeType</code>	string	The type of the script, either Unix Shell, Windows .BAT, or Windows VBScript. Allowed values: SH BAT VBS
<code>usageNotes</code>	string	Text that describes the script.
<code>changeLogComments</code>	string	Comments about the script for the change log.
<code>sharedScript</code>	boolean	Indicates whether or not the script is shared. If false, it is a "My Script" and can be accessed only by the user who created it.
<code>affectsServer</code>	boolean	Indicates whether or not this script modifies the managed server.

## Exceptions

`OpwareInvalidArgumentException` - is thrown if `scriptName`, `scriptText`, `codeType`, `usageNotes`, or `changeLogComments` is null or empty.

`jav.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.



## delete

Deletes a `Script` object.

### Result

SOAP Data Type: `string`

Description: The string `OK`.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptID</code>	<code>decimal</code>	The ID of the script to delete.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `scriptID` is null.

`OpwareObjectNotFoundException` - is thrown if no matching `Script` object is found.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

## execute

Runs a script on one or more managed servers.

To check the status of a script execution, invoke the `CommandSession.getDSESessionProgress` operation. To check the results of the script execution, invoke `CommandSession.getDSESessionResult` and `CommandSession.getDSESessionCommandResult`.

### Result

SOAP Data Type: `decimal`

Description: The ID of the Command Engine session handling the script execution. This ID is not the same as the `scriptID`, which identifies the `Script` object that has been saved in the Model Repository.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>dseArgument</code>	<code>DistributedScriptArgument</code>	The complex type that specifies the arguments for running the distributed script. The arguments include the <code>customerIDs</code> , the <code>scheduleDate</code> , and the <code>parameters</code> , a complex type that contains the <code>serverIDs</code> and the <code>scriptID</code> .

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `dseArgument` is null.

`OpwareInvalidArgumentException` - is thrown if `dseArgument.parameters.serverIDs` and `dseArgument.parameters.serverGroupIDs` are null or empty.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

## get

Retrieves a `Script` object, which represents a distributed script that can be executed on managed servers.

### Result

SOAP Data Type: `Script`

Description: The `Script` object of the specified ID.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptID</code>	<code>decimal</code>	The ID of the script to retrieve.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `scriptID` is null.

`OpswareObjectNotFoundException` - is thrown if no matching `Script` object is found.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

## getCurrentVersion

Retrieves the current version of a script as a `ScriptVersion` object.

### Result

SOAP Data Type: `ScriptVersion`

Description: The current `ScriptVersion` of the script specified.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptID</code>	<code>decimal</code>	The ID of the script which has the current version.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `scriptID` is null.

`OpwareObjectNotFoundException` - is thrown if no matching `Script` or `ScriptVersion` object is found.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

## getScriptText

Retrieves the text (source code) of a script's current version.

### Result

SOAP Data Type: `string`

Description: The text of the script specified.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptID</code>	<code>decimal</code>	The ID of the script which has the text.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `scriptID` is null.

`OpswareObjectNotFoundException` - is thrown if no matching `Script` object is found.

`jav.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

## update

Updates the name and usage notes of a `Script` object.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptID</code>	<code>decimal</code>	The ID of the script to update.
<code>scriptName</code>	<code>string</code>	The display name of the script. The <code>scriptName</code> must be unique and less than 100 characters in length.
<code>usageNotes</code>	<code>string</code>	Text that describes the script.

### Exceptions

`OpawareNullArgumentException` - is thrown if `scriptID` is null.

`OpawareInvalidArgumentException` - is thrown if `scriptName` or `usageNotes` is null or empty.

`OpawareObjectNotFoundException` - is thrown if no matching `Script` object is found.

`jav.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.



## updateScriptText

Updates the text (source code) of a `Script` object, creating a new version.

### Result

SOAP Data Type: `decimal`

Description: The ID of the new version of the script.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scriptID</code>	<code>decimal</code>	The ID of the script to update.
<code>scriptText</code>	<code>string</code>	The text (source code) of the script.
<code>changeLogComments</code>	<code>string</code>	Comments about the changes made to the script for this version.
<code>affectsServer</code>	<code>string</code>	Indicates whether or not this script modifies the managed server.

### Exceptions

`OpswareNullArgumentException` - is thrown if `scriptID` is null.

`OpswareNullArgumentException` - is thrown if `scriptText` or `changeLogComments` is null or empty.

`OpswareObjectNotFoundException` - is thrown if no matching `Script` object is found.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

# Chapter 7: Facility Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- create
- decommission
- get
- getIDs
- getList
- update

In Opsware SAS, a facility refers to the collection of servers and server groups that are managed by a single Model Repository. Typically, a facility corresponds to a specific data center. In an Opsware multimaster mesh, users can manage servers in any facility from the Opsware Command Center. When a user updates data in a facility, the Model Repository for that facility is synchronized with the other Model Repositories in the mesh. In the Opsware Command Center, a facility is identified by a customer-assigned facility name and an Opsware-assigned facility ID.

## create

Creates a new facility.

### Result

SOAP Data Type: decimal

Description: The ID (primary key) of the new facility.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
facilityName	string	The short name of the new facility. The value must be in uppercase and cannot contain spaces.
displayName	string	The name of the new facility displayed in the Opsware Command Center.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `facilityName` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## decommission

Removes a facility from use. In the Opsware Command Center, this action is called “deactivate.”

When you deactivate a facility, the facility is still listed in the Opsware Command Center; however, it is marked with the letter D. The facility short name cannot be re-used, even if the facility is deactivated.

### Result

SOAP Data Type: `string`

Description: The result of the operation.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>facilityID</code>	<code>string</code>	The ID of the facility to be decommissioned (deactivated).

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `facilityID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## get

Returns the facility specified by the ID

### Result

SOAP Data Type: `Facility`

Description: The object that represents the facility.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>facilityID</code>	<code>decimal</code>	The ID of the facility to retrieve.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `facilityID` is null.

`OpswareObjectNotFoundException` - is thrown if no facility object matches the specified `facilityID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getIDs

Returns an array of facility IDs that matches the criteria specified by the `facilityFilter` argument. To get all facilities, specify a null `facilityFilter` (or null elements of `facilityFilter`). To get specific active facilities, specify the `facilityFilter.facilityIDs`. To search for all facilities associated with a customer, specify the `facilityFilter.customerID`.

### Result

SOAP Data Type: `decimal []`

Description: An array of facility IDs

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>facilityFilter</code>	<code>FacilityFilter</code>	The object that specifies the search criteria for this operation.

### Exceptions

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered

## getList

Returns a list of facilities that matches the criteria specified by the `facilityFilter` argument. This operation retrieves facilities in one of the following ways: To get all facilities, specify a null `facilityFilter`. To search for active facilities by ID, specify `facilityFilter.facilityIDs`. To search for all facilities associated with a customer, specify the `facilityFilter.customerID`.

### Result

SOAP Data Type: `FacilityList`

Description: An object that represents multiple facilities.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>facilityFilter</code>	<code>FacilityFilter</code>	The object that specifies the search criteria for this operation.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## update

Edits facility's attributes.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>facilityID</code>	<code>decimal</code>	The ID of the facility to be updated. The ID itself cannot be changed.
<code>facilityName</code>	<code>string</code>	The short name of the facility. The value must be in uppercase and cannot contain spaces.
<code>displayName</code>	<code>string</code>	The name of the facility displayed by the Opsware Command Center.
<code>status</code>	<code>string</code>	The facility's status, either <code>ACTIVE</code> or <code>DECOMMISSIONED</code> .

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `facilityID` or `facilityName` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

# Chapter 8: Node Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- attachServer
- create
- delete
- detachCustomers
- detachPlatforms
- detachServer
- detachSoftwarePackage
- get
- getChild
- getChildList
- getCustomerList
- getDescendant
- getIDs
- getInheritedSoftwarePackageList
- getList
- getNodeList
- getParent
- getPlatformList
- getServerList
- getSoftwarePackageIDOverrideTupleList
- getSoftwarePackageList
- move
- setCustomers
- setPlatforms
- setSoftwarePackages

In Opware SAS, a node is an element in a tree that organizes Opware entities and enables inheritance. For example, the hierarchical structure of the tree enables software application nodes to inherit properties from other software application nodes. An Opware entity is an object such as `Customer`, `Facility`, `Software Package`, and `Server`. Behind the scenes, Opware SAS stores these entities in the Model Repository database. To specify relationships, you can attach entities to nodes. For example, you can attach servers, software packages, customers, platforms to a Custom Application node.

The node tree is made up of subtrees called stacks. Some of these stacks are for internal use by Opware SAS, and others, such as the System Utilities stack, can be manipulated by end users in the Opware Command Center.

## attachServer

Attaches a server to a node.

### Result

SOAP Data Type: `string`

Description: Text message describing the results of attaching a server to a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	The ID of the node to be associated with the server.
<code>serverID</code>	<code>decimal</code>	The ID of the server to be associated with the node.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` or `serverID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareObjectNotFoundException` - is thrown if the node or server is not found

## create

Creates a new node in any of the any of the six Application stacks, which include the Application Servers, DatabaseServer, OS Extras, Other Applications, System Utilities, and WebServer stacks. The position in the tree where the node is created is determined by the `parentNodeID`, which is the first parameter of the create operation. The created node will become a child of the specified parent node.

### Result

SOAP Data Type: `decimal`

Description: ID of the new node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>parentNodeID</code>	<code>decimal</code>	The parent node ID under which you want to create a new node.
<code>shortName</code>	<code>string</code>	A short name for the new node. The value must be all upper case and contain no spaces.
<code>description</code>	<code>string</code>	A brief description of the new node.
<code>notes</code>	<code>string</code>	Notes that describe the node you are creating.
<code>allowDevice</code>	<code>boolean</code>	Specify if you want to allow devices (servers) to be attached to the new node.
<code>blockedNode</code>	<code>boolean</code>	Specify if you want the node to be locked. If a node is locked, it cannot be moved or edited except by a user who has the appropriate privileges.

## Exceptions

`OpswareNullArgumentException` - is thrown if the specified `parentNodeID`, or `shortName` is null or empty.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## delete

Deletes a node from the tree.

### Result

SOAP Data Type: `string`

Description: Text message describing the results of deleting a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node to delete.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## detachCustomers

Detaches a list of customers from a node.

### Result

SOAP Data Type: `string`

Description: Text message describing the results of detaching a list of customers from a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node to detach from customers.
<code>customerIDs</code>	<code>decimal []</code>	An array of customer IDs to detach from a node.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` or `customerIDs` is null.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## detachPlatforms

Detaches a list of platforms (operating systems) from a node.

### Result

SOAP Data Type: `string`

Description: A text message describing the results of detaching a list of platforms from a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node to detach from the platforms.
<code>platformIDs</code>	<code>decimal []</code>	An array of platform IDs to detach from a node.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` or `platformIDs` is null.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## detachServer

Detaches a server from a node.

### Result

SOAP Data Type: `string`

Description: Text message describing the results of detaching a server from a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	The ID of the node to detach servers.
<code>serverID</code>	<code>decimal</code>	The ID of the server to detach from the node.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` or `serverID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareObjectNotFoundException` - is thrown if the node or server is not found

## detachSoftwarePackage

Detaches a software package from a node.

### Result

SOAP Data Type: `string`

Description: A text message describing the results of detaching a software package from a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node to detach from the software package.
<code>softwarePackageID</code>	<code>decimal</code>	ID of the software package to detach from the node.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `nodeID` or `softwarePackageID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareObjectNotFoundException` - is thrown if the node or package is not found

## get

Gets a `Node` object specified by the node ID. A `Node` object contains information such as its ID (unique identifier), short name, and parent ID.

### Result

SOAP Data Type: `Node`

Description: A `Node` object.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
nodeID	decimal	ID of the node to retrieve.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getChild

Returns the `Node` object for the immediate descendant (specified by short name) of the node specified by ID. A `Node` object contains information such as its ID (unique identifier), short name, and parent ID.

### Result

SOAP Data Type: `Node`

Description: A `Node` object.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the parent node whose child you want to get.
<code>childShortName</code>	<code>string</code>	The internal child node name used by Opsware SAS. The value must be all upper case and contain no spaces.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if the parent or child `Node` object is not found.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getChildList

Gets a list of children node objects (the immediate descendents) that belong to a specified parent node.

### Result

SOAP Data Type: `NodeList`

Description: A list of `Node` objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the parent node whose children you want to get.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if the parent or child `Node` object is not found.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getCustomerList

Gets a list of `Customer` objects that are associated with a node. A `Customer` object represents a customer entity in Opsware SAS, and consists of such attributes as a customer ID, customer name, display name, status, authorization domain, and so on.

### Result

SOAP Data Type: `CustomerList`

Description: A list of `Customer` objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node whose customer list you want to get.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## getDescendant

Returns the `Node` object for the descendant (specified by short name path) of the node specified by node ID.

### Result

SOAP Data Type: `Node` object.

Description: A `Node` object contains information such as its ID (unique identifier), short name, and parent ID.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node whose descendent you want to get.
<code>descendantShortNamePath</code>	<code>java.lang.String []</code>	An array of strings representing the shortpath name of the descendent nodes you are getting.

### Exceptions

`OpawareNullArgumentException` - is thrown if the specified `nodeID` or `descendantShortNamePath` is null.

`OpawareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`, or if no descendant `Node` object is found.

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getIDs

Gets a list of node IDs. Each node in Opsware SAS contains a unique ID to distinguish it from all other nodes.

### Result

SOAP Data Type: decimal []

Description: An array of node IDs.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeFilter</code>	<code>nodeFilter</code>	Determines the search results of the <code>getIDs</code> operation.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeFilter` is null.

`OpswareInvalidArgumentException` - is thrown if all the fields of `nodeFilter` are null, or `nodeFilter.nodeIDs` is a non-empty list.

`OpswareListRetrievalErrorException` - is thrown if an error is encountered while retrieving the list of IDs.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareObjectNotFoundException` - is thrown if no `Node` object is found

## getInheritedSoftwarePackageList

Gets a list of `SoftwarePackage` objects (software packages) that a node inherits from its parents and ancestors in the tree.

### Result

SOAP Data Type: `SoftwarePackageList`

Description: A list of `SoftwarePackage` objects, which represent all packages that the specified node inherits from its parents and ancestors in the tree.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node from which you want to get a list of attached software packages.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpwareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getList

Gets a list of Node objects.

### Result

SOAP Data Type: `NodeList`

Description: A list of Node objects. A Node object contains information such as its ID (unique identifier), short name, and parent ID.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeFilter</code>	<code>NodeFilter</code>	Determines the search results of the <code>getList</code> operation.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeFilter` is null.

`OpswareInvalidArgumentException` - is thrown if all the fields of `nodeFilter` are null.

`OpswareListRetrievalErrorException` - is thrown if an error is encountered while retrieving the list of objects.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getNodeList

Gets a list of `Node` objects that are associated with a specified server.

### Result

SOAP Data Type: `NodeList`

Description: A list of `Node` objects. A `Node` object contains information such as its ID (unique identifier), short name, and parent ID.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	ID of the server with the <code>Node</code> objects retrieved.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpwareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpwareListRetrievalErrorException` - is thrown if an error is encountered while retrieving the list of objects.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getParent

Returns the parent Node object for the node specified by ID.

### Result

SOAP Data Type: Node

Description: A Node object. A Node object contains information such as its ID (unique identifier), short name, and parent ID.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
nodeID	decimal	ID of the node whose parent you want to get.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if no Node object matches the specified `nodeID`, or if no parent Node object is found.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getPlatformList

Gets a list of `Platform` objects of a specified node. A `Platform` object represents the definition of an operating system in Opware SAS. A platform is the name and version of an operating system (for example, Red Hat Linux 4.2). Each `Platform` object contains varied elements such as the hardware architecture associated with the OS, platform ID, OS version, platform name, and so on.

### Result

SOAP Data Type: `PlatformList`

Description: A list of `Platform` objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node from which you want to get a list of platform objects.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpwareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getServerList

Gets a list of `Server` objects associated with a node. A `Server` object represents a server machine managed by Opsware SAS, and contains such elements as a machine ID, OS version, serial number, server ID, and so on.

### Result

SOAP Data Type: `ServerList`

Description: A list of `Server` objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node from which you want to get a list of platform objects.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## getSoftwarePackageIDOverrideTupleList

Gets a list of `SoftwarePackageIDOverrideTuple` objects. This list indicates the installation order of inherited software packages.

### Result

SOAP Data Type: `SoftwarePackageIDOverrideTupleList`

Description: A tuple in an ordered list that indicates the installation order of a node's inherited software packages.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node from which you want to get a list of <code>SoftwarePackageIDOverrideTuple</code> objects.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

## getSoftwarePackageList

Gets a list of `SoftwarePackage` objects. A software package represents an installable chunk of software, such as an RPM.

### Result

SOAP Data Type: `SoftwarePackageList`

Description:

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	The ID of the node from which you want to get a list of <code>SoftwarePackage</code> objects.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` is null.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## move

Moves a node from one location to another. When you move a node to a new destination node, the moved node becomes a child of the destination node. (The destination node becomes the parent node.) When you move a node, all of that node's dependencies and all servers attached to that node remain intact. All other attributes of the node are inherited from the new parent and all direct attachments remain intact.

Some restrictions on moving a node:

- You can only move nodes within the following Application stacks: Application Servers, DatabaseServer, OS Extras, Other Applications, System Utilities, and WebServer.
- You cannot move a node to itself or to its child.
- You cannot move a locked node to a destination node that is not locked, but you can move an unlocked node to a locked node. If you want to move a locked node to another locked node, you need locking permissions to do so.
- You cannot move a top-level node (such as Application Servers, OS Extras, Web Servers, and so on).
- You can only move a node within its current stack.

## Result

SOAP Data Type: `string`

Description: A message describing the results of moving a node.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
nodeID	decimal	ID of the node you are moving.
newParentNodeID	decimal	ID of the parent under which you want to move the node.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `nodeID` or `newParentNodeID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## setCustomers

Associates a list of customers with a `Node` object. A `Customer` object represents a customer entity in Opware SAS, and consists of such entities as a customer ID, customer name, display name, status, authorization domain, and so on.

The `Customer` object determines who can view and edit the node (with the appropriate customer permissions). After using this operation, the existing list of associated customers will be overwritten.

### Result

SOAP Data Type: `string`

Description: A message describing the result of associating the customers with the `Node`.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>nodeID</code>	<code>decimal</code>	ID of the node you are associating with customers.
<code>customerIDs</code>	<code>decimal []</code>	An array of customer IDs to associate with a node.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `nodeID` or `customerIDs` is null.

`OpwareInvalidArgumentException` - is thrown if the specified list of `customerIDs` contains more than one ID and the `Node` specified by `nodeID` is not in the service level stack.

`OpwareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## setPlatforms

Associates a list of platforms (operating systems) with a node. (For a complete list of supported operating systems, refer to the *Opware® SAS 5.2 User's Guide*.) After using this operation, the existing list of associated platforms will be overwritten.

### Result

SOAP Data Type: `string`

Description: A message describing the result of associating the platforms with the Node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
nodeID	decimal	ID of the node you are associating platforms with.
platformIDs	decimal	An array of platform IDs to associate with a node.

## Exceptions

`OpswareNullArgumentException` - is thrown if the specified `nodeID` or `platformIDs` is null.

`OpswareInvalidArgumentException` - is thrown if the specified list of `platformIDs` contains more than 1 one ID and the `Node` specified by `nodeID` is not in the service level stack.

`OpswareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## setSoftwarePackages

Attaches software packages to a node. A software package represents an installable chunk of software, such as an RPM. This operation overwrites the existing list of attached software packages. This operation assumes that the list of packages (`SoftwarePackageIDOverrideTuple []`) is ordered in the sequence required for installation.

### Result

SOAP Data Type: string

Description: A message describing the results of attaching software packages to a node.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
nodeID	decimal	ID of the node to which you want to attach a software package.
softwarePackageIDOverrideTuples []	softwarePackageIDOverrideTuples	An array of <code>softwarePackageIDOverrideTuples</code> objects. The array is an ordered list that indicates the installation order of inherited software packages.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `nodeID` or `softwarePackageIDOverrideTuples` argument is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareObjectNotFoundException` - is thrown if no `Node` object matches the specified `nodeID`.

## Chapter 9: Platform Web Service

### IN THIS CHAPTER

This chapter describes the following SOAP operations:

- get
- getIDs
- getList

In Opware SAS, the term platform refers to an operating system. Each platform has a unique identifier (ID) and a hard-coded name, both assigned by Opware SAS. Although users can assign a display name to an operating system in the Opware Command Center, the operations in this chapter refer only to the IDs and names assigned by Opware SAS.

## get

Returns the Platform object of the specified ID.

Platform IDs are assigned by Opsware SAS and are not visible in the Opsware Command Center.

## Result

SOAP Data Type: Platform

Description: The specific Operating System, such as RedHat Linux AS 2.1.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
platformID	decimal	The ID of the platform to be returned.

## Exceptions

OpswareNullArgumentException - is thrown if the specified platformID is null.

OpswareObjectNotFoundException - is thrown if no Platform object matches the specified platformID.

OpswareException - is thrown when an application level error is encountered.

OpswareInternalErrorException - is thrown when an internal error is encountered.

OpswareAccessControlException - is thrown when the requested access is denied.

OpswareAuthenticationException - is thrown when user authentication fails.

java.rmi.RemoteException - is thrown when a framework or network error is encountered.

## getIDs

Returns an array of platform IDs that matches the criteria specified by the `platformFilter` argument. To get all platforms, specify a null `platformFilter` (or null elements of `platformFilter`). To get specific platforms, specify the `platformFilter.platformIDs`. To search for a platform by name, specify the `platformFilter.platformName`.

### .Result

SOAP Data Type: `decimal []`

Description: An array of IDs of `Platform` objects.

### Argument

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>platformFilter</code>	<code>PlatformFilter</code>	The object that specifies the search criteria for this operation.

### Exceptions

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getList

Returns a list of Platform objects that matches the criteria specified by the platformFilter argument.

The search performed by this operation follows the same criteria specified by the platformFilter argument documented for getIDs.

### Result

SOAP Data Type: PlatformList

Description: An object containing a list of Platform objects.

### Argument

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
platformFilter	PlatformFilter	The object that specifies the search criteria for this operation.

### Exceptions

OpswareException - is thrown when an application level error is encountered.

OpswareInternalErrorException - is thrown when an internal error is encountered.

OpswareAccessControlException - is thrown when the requested access is denied.

OpswareAuthenticationException - is thrown when user authentication fails.

java.rmi.RemoteException - is thrown when a framework or network error is encountered.

# Chapter 10: Scheduling Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- cancel
- schedule
- update

The Scheduling Web Service enables you to schedule the following functions :

- Snapshot
- Audit
- Application configuration push to servers

For example, you could create a snapshot template with the OCC Client and then use the Scheduling Web Service to create the snapshot at 2:00 AM every Monday. For more information on snapshots, audits, and application configuration management, see the *Opware<sup>®</sup> SAS 5.2 User's Guide*.

## cancel

Cancels the schedule of a function (snapshot, audit, application configuration push). This operation removes the schedule, but does not remove the snapshot template, audit template, or application configuration.

### Result

SOAP Data Type: `string`

Description: The result of this operation.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scheduleID</code>	<code>decimal</code>	The identifier of the schedule to be cancelled.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareNullArgumentException` - is thrown if `scheduleID` is null

`OpswareAppConfigException` - is thrown if an Opsware audit or snapshot error is encountered



## schedule

Schedule a previously defined function (snapshot, audit, or application configuration push). This operation does not create the snapshot template, audit template, or application configuration. You create these objects with the OCC Client.

### Result

SOAP Data Type: `decimal`

Description: The unique identifier of the schedule created by this operation.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>operationType</code>	<code>string</code>	<p>The type of function to be scheduled. Allowed values:</p> <ul style="list-style-type: none"> <li><code>Snapshot</code></li> <li><code>ComplianceTest</code></li> <li><code>AuditConfig</code></li> <li><code>VisualPackage</code></li> </ul> <p>The <code>Snapshot</code> type corresponds to a snapshot (of the Server Compliance feature) in the OCC Client. The <code>ComplianceTest</code> type corresponds to an audit (of the Server Compliance feature) in the OCC Client. The <code>AuditConfig</code> type corresponds to an application configuration push in the OCC Client. The <code>VisualPackage</code> type is for an object used internally by Opware SAS and is not exposed by the OCC Client.</p>

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
operationID	decimal	The identifier of the snapshot or audit template created in the OCC Client. This argument is not used if the operationType is set to AuditConfig.
auditConfigTargetServers	decimal []	An array of identifiers of the servers to which the application configuration is pushed. This argument is used only if the operationType is set to AuditConfig.
notificationSpec	EmailNotificationSpec	The object that specifies the "To:" and "Reply-To:" values in the header of the email notification sent when the function finishes. If the notificationSpec is null, then the email notification is not sent.
scheduleSpec	ScheduleSpec	The object that specifies when to run the scheduled function.

## Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareNullArgumentException` - is thrown if `scheduleSpec` is null, or if `operationType` is null or empty,

`OpwareAppConfigException` - is thrown if an Opware audit or snapshot error is encountered

## update

Modifies the schedule information of the function (snapshot, audit, application configuration push).

### Result

SOAP Data Type: `string`

Description: The result of this operation.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>scheduleID</code>	<code>decimal</code>	The identifier of the schedule to be updated.
<code>scheduleSpec</code>	<code>ScheduleSpec</code>	The object that specifies when to run the scheduled function.
<code>notificationSpec</code>	<code>EmailNotificationSpec</code>	The object that specifies the "To:" and "Reply-To:" values in the header of the email notification sent when the function finishes. If the <code>notificationSpec</code> is null, then the email notification is not sent.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareNullArgumentException` - is thrown if `scheduleID` or `scheduleSpec` is null

`OpwareAppConfigException` - is thrown if an Opware audit or snapshot error is encountered



# Chapter 11: Security Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- assignAdministratorRole
- assignCustomerPermissions
- assignFacilityPermissions
- assignUsers
- createSecurityRole
- createUser
- getAssignedSecurityRoleNameList
- getCustomerPermissionList
- getFacilityPermissionList
- getSecurityRoleNameList
- getUsernameList
- isAssignedAdministratorRole
- removeSecurityRole
- removeUser
- unassignAdministratorRole
- unassignUsers
- updatePassword

Opware SAS enforces a security policy that allows only authorized users to perform specific operations on specific servers. Opware users belong to security roles (also called user groups). If an Opware user is assigned to more than one security role, that user has the permissions provided by the union of the multiple security roles.

A security role is a set of permissions, of which there are two types: resources and features. A resource is usually a set of managed servers or software nodes. A resource permission specifies if the users in a role can view or modify a resource. Resources are categorized by customers, facilities, server groups, and the node stack. The permissions assigned to a resource specify READ, WRITE, or NONE access. An Opware feature is a task, such as running a script or uploading a patch. Feature permissions, define the tasks that can be performed by the users of a role. Resource permissions specify the objects (software or servers) that can be modified or viewed with a feature. To use a feature on a resource, the user must belong to a role that has the necessary permissions for both the feature and resource. For more information, see the “User and Group Setup” chapter of the *Opware® SAS 5.2 User's Guide*.

Except for the `updatePassword` operation, all operations in the Security Web Service must be called by a user with the administrative privilege. (The default user with this privilege is the `admin` user.) The user name and password of this administrative user are required as the first parameters of these operations.



## assignAdministratorRole

Assigns a specified Opsware user to the Opsware administrator (`admin`) security role.

The Opsware administrator can create Opsware users and assign them to a security role.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.

### Exceptions

`OpawareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## assignCustomerPermissions

Assigns customer permissions to a security role. Customer permissions specify which customers are available to Opsware users in the associated security role. In addition, these permissions specify whether Opsware users have read-only or read/write access to the servers associated with a customer.

As an Opsware administrator, you create a security role and then assign access permissions to that security role. After you assign permissions, you assign Opsware users to the security role.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of a security role to which the customers listed will be added.
<code>customerIDs</code>	<code>decimal []</code>	An array of customer IDs whose permissions are to be changed.
<code>accessList</code>	<code>string []</code>	An array of strings. Allowed values are: READ, WRITE, NONE.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## assignFacilityPermissions

Assigns facility permissions to a security role. Facility permissions specify which facilities are available to Opsware users in the associated security role. In addition, these permissions specify whether Opsware users have read-only or read/write access to the servers associated with a facility.

As an Opsware administrator, you create a security role and then assign access permissions to that security role. After you assign permissions, you assign Opsware users to the security role.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of a security role to which the facilities listed will be added.
<code>facilityIDs</code>	<code>decimal []</code>	An array of facility IDs whose permissions are to be changed.
<code>accessList</code>	<code>string []</code>	An array of strings. Allowed values are: READ, WRITE, NONE.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpawareInternalErrorException` - is thrown when an internal error is encountered.

`OpawareAccessControlException` - is thrown when the requested access is denied.

`OpawareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## assignUsers

Assigns Opsware users to a security role. In Opsware SAS, you assign permissions to a security role and then assign Opsware users to that security role. If an Opsware user is assigned to more than one security role, that user has the set of permissions formed by the union of the multiple security roles.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of a security role to which the users will be assigned.
<code>accessList</code>	<code>string []</code>	An array of user names to be assigned to the security role.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## createSecurityRole

Creates an empty security role. After creating a security role, the Opsware administrator assigns users to the security role.

In addition to the groups that the Opsware administrator creates, Opsware SAS comes with pre-defined groups: Basic, Intermediate, and Advanced. An Opsware administrator can assign a user to one or more security roles.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of the security role to be created.
<code>description</code>	<code>string</code>	The description of the security role to be created.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.



`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## createUser

Creates a new Opsware user. As an Opsware administrator, you create a security role and then assign access permissions to that security role. After you assign permissions, you assign Opsware users to that security role. If an Opsware user is assigned to more than one security role, that user has the permissions provided by the union of the multiple security roles.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>userName</code>	<code>string</code>	The login name of the new Opsware user.
<code>userPassword</code>	<code>string</code>	The login password of the new Opsware user.
<code>firstName</code>	<code>string</code>	The first name of the new Opsware user.
<code>lastName</code>	<code>string</code>	The last name of the new Opsware user.
<code>emailAddress</code>	<code>string</code>	The email address of the new Opsware user.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getAssignedSecurityRoleNameList

Returns a list of names of all the security roles ( user groups) assigned to an Opsware user.

### Result

SOAP Data Type: `string []`

Description: An array of security role names.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>userName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>password</code>	<code>string</code>	The password of an Opsware user with administrative privileges.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getCustomerPermissionList

Returns a list of all the customer IDs assigned to the specified security role.

### Result

SOAP Data Type: `permissionList`

Description: An array of `permission` objects, which specify read and write access to servers associated with the customers.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of the security role of the permissions to retrieve.

### Exceptions

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getFacilityPermissionList

Returns a list of all the facility IDs assigned to the specified security role.

### Result

SOAP Data Type: `permissionList`

Description: An array of `permission` objects, which specify read and write access for customers.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of the security role of the permissions to retrieve.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getSecurityRoleNameList

Returns a list of names of all the security roles (user groups) that exist.

### Result

SOAP Data Type: `string []`

Description: An array of security role names.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getUserNameList

Returns a list of all Opsware user names that have been assigned to a specified security role.

### Result

SOAP Data Type: `string []`

Description: An array of Opsware user names.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of the security role of the retrieved users.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## isAssignedAdministratorRole

Indicates whether or not the specified Opsware user has been assigned to the Opsware security role.

### Result

SOAP Data Type: boolean

Description: The result of the test performed by this operation.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
adminUserName	string	The name of an Opsware user with administrative privileges.
adminPassword	string	The password of an Opsware user with administrative privileges.
userName	string	The name of the Opsware user that this operation checks for administrator privileges.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## removeSecurityRole

Removes a security role (user group) along with its permission settings.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of the security role to be removed.

### Exceptions

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## removeUser

Removes the specified Opsware user.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>userName</code>	<code>string</code>	The name of the Opsware user to be removed.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## unassignAdministratorRole

Unassigns specified Opsware users from the Opsware administrators security role.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>userNames</code>	<code>string []</code>	An array of Opsware user names to be removed from the administrator role.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## unassignUsers

Unassigns users from the specified security role.

### Result

SOAP Data Type: `string`

Description: The string OK.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>adminUserName</code>	<code>string</code>	The name of an Opsware user with administrative privileges.
<code>adminPassword</code>	<code>string</code>	The password of an Opsware user with administrative privileges.
<code>roleName</code>	<code>string</code>	The name of the security role from which Opsware users are to be unassigned.
<code>userNames</code>	<code>string []</code>	An array of Opsware user names to be unassigned from a security role.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## updatePassword

Changes the password for a given Opsware user.

### Result

SOAP Data Type: `string`

Description: The string `true` if the password change was successful.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>userName</code>	<code>string</code>	The name of the Opsware user whose password is to be changed.
<code>oldPassword</code>	<code>string</code>	The Opsware user's current password.
<code>newPassword</code>	<code>string</code>	The Opsware user's new password.

### Exceptions

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

# Chapter 12: Server Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- associateWithCustomer
- delete
- get
- getAgentStatus
- getChangeLogList
- getComponents
- getComponentsList
- getEventList
- getIDs
- getList
- getReachableServer
- reconcile
- update

The Server Web Service allows you to perform a variety of operations on an Opware managed server, such as associating the server with a customer, checking the status of the Opware Agent, getting a list of changes for a server, and reconciling the server.

## associateWithCustomer

Associates a server with a customer. A `Customer` object represents a customer entity in Opsware SAS, and consists of such entities as a customer ID, customer name, display name, status, authorization domain, and so on.

### Result

SOAP Data Type: `string`

Description: A text message that indicates the result of associating a server with a customer (success or failure).

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server to be associated with the customer.
<code>customerID</code>	<code>decimal</code>	The ID of the customer to be associated with the server.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID` or the `customerID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## delete

Deletes the specified server from the Model Repository database. When a server is deleted from the Model Repository, it is no longer recognized by Opsware SAS.

### Result

SOAP Data Type: `string`

Description: A text string that indicates the server has been deleted.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server to be deleted.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## get

Gets the `Server` object specified by the server ID.

### Result

SOAP Data Type: `Server`

Description: An object that contains about 30 attributes, such as the server ID, IP address, Agent status, MID, and so on. Each server is identified in Opsware SAS by a unique primary ID.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server to get.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getAgentStatus

Gets the status of the Opsware Agent on the managed server specified by server's ID. A server is monitored by Opsware SAS through the Agent. If the Agent is unreachable, then the server is no longer being managed – in other words, Opsware SAS cannot access the server. This operation allows you to check the reachability of the Agent on the specified server.

### Result

SOAP Data Type: `string`

Description: Returns a message on the current state of the Agent: either OK or unreachable.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server which has the Agent you want to get the status of.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getChangeLogList

Returns a list of activity log items for the specified server, during a time window specified by the start and end time-points. The activity log notes then changes made to a server, such as changing its name or its network information, adding or deleting attributes, adding or removing a server from a node, running a communication test against the server, running a reconcile on the server, and so on. This information is stored in the change log list (also referred to as “Server History”) and is retrievable using this operation.

### Result

SOAP Data Type: `ServerChangeLogList`

Description: A list that includes any changes made to the server during the specified time period.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server from which you want to get a change log list.
<code>start</code>	<code>dateTime</code>	The start time of the period you want to retrieve for the server change log list.
<code>end</code>	<code>dateTime</code>	The end time of the period you want to retrieve for the server change log list.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID`, `start`, or `end` is null.

`OpswareObjectNotFoundException` - is thrown if no change log item is found for the server specified.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getComponents

Gets the components of a `Server` object specified by its ID. Components of a server include the `Customer` object that owns the server, the `Facility` object in which the server is located, a list of `CpuComponent` objects for all the CPU's on that server, a list of `MemoryComponents`, a list of `StorageComponents`, a list of `InterfaceComponents`, a list of service-level `Node` objects, and so on.

### Result

SOAP Data Type: `ServerComponents`

Description: `Server` object components from the specified server including CPU, storage, memory, interface, service level, customer, facility, and so on.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server from which you want to get the components.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpswareObjectNotFoundException` - is thrown if no server object matches the specified `serverID`.

## getComponentsList

Gets a list of `ServerComponents` objects. Components of a server include the `Customer` object that owns the server, the `Facility` object in which the server is located, a list of `CpuComponent` objects for all the CPU's on that server, a list of `MemoryComponents`, a list of `StorageComponents`, a list of `InterfaceComponents`, a list of service-level `Node` objects, and so on.

The maximum number of `ServerComponents` objects that can be retrieved at a time is 200. This limit is enforced to avoid memory issues because of the size of the `ServerComponents` object. To retrieve more than 200 objects, a client application would get the list of server IDs and then iterate through the list, retrieving of 200 `ServerComponents` objects at a time.

This operation uses the same search criteria and algorithm as the `getList` operation.

### Result

SOAP Data Type: `ServerComponentsList`

Description: A list of `ServerComponent` objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverFilter</code>	<code>ServerFilter</code>	The object that specifies the search criteria for this operation.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `serverFilter` is null.

`OpwareInvalidArgumentException` - is thrown if all the fields of `serverFilter` are null or empty.

`OpwareTooManyItemsException` - is thrown if the requested number of objects is greater than 200.

`OpwareListRetrievalErrorException` - is thrown if an error is encountered while retrieving the list of objects.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## getEventList

Gets a list of server events for `Server` objects that have been added, modified, or removed since a specified time.

The Server Web Service maintains a sliding window (with a default size of two hours) of events describing changes to `Server` objects. This window makes it possible for you to update a client-side cache of `Server` objects without having to retrieve all of the objects. For example, you could maintain a client-side cache as follows:

- 1** Initially, invoke the `getList` operation to retrieve all relevant `Server` objects.
- 2** At specified time intervals within the two hour time window, perform these steps:
  - invoke the `getEventList` operation to retrieve the events for the `Server` objects that have been changed.
  - Update the client-side cache with the information returned by `getEventList`.

If you do not update the client-side cache for more than two hours, a full retrieval will be required to obtain all changes. You can set the size of the sliding window for optimal performance by changing the Web Services Data Access Engine configuration file. (For instructions on changing the configuration file, see the *Opware® SAS 5.2 Administration Guide*.)

There are three types of server events: `ADDED`, `MODIFIED`, or `REMOVED`. If an event is of type `ADDED` or `MODIFIED`, the `ServerEvent` object embeds the `Server` object. If the event is of type `REMOVED`, the `ServerEvent` object embeds the server ID.

Note: The list of `ServerEvent` objects returned may contain an `ADDED` event, a `MODIFIED` event, as well as a `REMOVED` event for the same server. However, duplicate items for events of the same type are removed. Therefore, the list can contain only one `MODIFIED` event for a server, even if the server has been modified several times within the two hour time window.

### Result

SOAP Data Type: `ServerEventList`

Description: A list of server events (additions, modifications, deletions) from the cache (sliding window) maintained by the Server Web Service.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverFilter</code>	<code>ServerFilter</code>	The object that specifies the search criteria for this operation. <code>ServerFilter.lastRequestTimeStamp</code> must be specified. Other <code>ServerFilter</code> elements are ignored.

## Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverFilter` is null.

`OpswareInvalidArgumentException` - is thrown if the `serverFilter.lastRequestTimeStamp` is null, is after the current time, or is prior to the current cache sliding window start time.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getIDs

Returns an array of server IDs that matches the criteria specified by the `serverFilter` argument.

To get all servers, specify a null `serverFilter`. To search for a single server, specify either the `serverHostname`, `serverIPAddress`, or `machineID` in the `serverFilter`. To search for multiple servers, specify the `serverFilter` elements listed in the last `else` clause of the below algorithm. For example, to search for servers associated with certain customers, specify `serverFilter.customerIDs`.

This operation searches for servers with the following algorithm:

```

if serverFilter is null
return all server IDs
else if server.serverHostname is not null
return server ID by name
else if serverFilter.serverIPAddress is not null
return server ID by IP
else if serverFilter.machineID is not null
return server ID by MAC
else
return server IDs by the conjunction (logical AND) of the
following serverFilter elements:
serverIDs
customerIDs
facilityIDs
agentStatusFilters
deploymentStageFilters
serverUseFilters

```

### Result

SOAP Data Type: `decimal []`

Description: An array of server IDs.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverFilter</code>	<code>ServerFilter</code>	The object that specifies the search criteria for this operation.

## Exceptions

`OpwareListRetrievalErrorException` - is thrown if an error is encountered while retrieving the list of objects.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getList

Returns a list of `Server` objects that matches the criteria specified by the `serverFilter` argument.

To search for a single server, specify either the `serverHostname`, `serverIPAddress`, or `machineID` in the `serverFilter`. To search for multiple servers, specify the `serverFilter` elements listed in the last `else` clause of the below algorithm. For example, to search for servers associated with certain customers, specify `serverFilter.customerIDs`.

This operation searches for servers with the following algorithm:

```

if serverFilter is null
  throw an exception
else if server.serverHostname is not null
  return server by name
else if serverFilter.serverIPAddress is not null
  return server by IP
else if serverFilter.machineID is not null
  return server by MID
else
  return the servers that match the conjunction (logical AND)
  of the following serverFilter elements:
  serverIDs
  customerIDs
  facilityIDs
  agentStatusFilters
  deploymentStageFilters
  serverUseFilters

```

### Result

SOAP Data Type: `ServerList`

Description: An array of `Server` objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverFilter</code>	<code>ServerFilter</code>	The object that specifies the search criteria for this operation.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `serverFilter` is null.

`OpwareInvalidArgumentException` - is thrown if all the fields of `serverFilter` are null or empty.

`OpwareListRetrievalErrorException` - is thrown if an error is encountered while retrieving the list of objects.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered

## getReachableServer

Gets the reachable `Server` object that matches the specified IP address. A server is reachable if the Opsware core can communicate with the Opsware Agent installed on the server.

### Result

SOAP Data Type: `Server`

Description: An object that contains about 30 attributes, such as the server ID, IP address, Agent status, MID, and so on. If the server is reachable, this operation will get the specified `Server` object.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>ipAddress</code>	<code>string</code>	IP address of the server.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `ipAddress` is null.

`OpwareInvalidArgumentException` - is thrown if the specified `ipAddress` is an empty string.

`OpwareObjectNotFoundException` - is thrown if no server matches the IP address specified, or if more than one server matches the IP address specified and none of them has an Agent status of OK, or if more than one server matches the IP address specified and more than one of these servers has an Agent status of OK.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## reconcile

Starts a reconcile session and returns the reconcile session ID. When you reconcile a server, you are updating the actual software configuration of a server based on the specified configuration stored in the Model Repository.

### Result

SOAP Data Type: `decimal`

Description: The reconcile session ID (if successful).

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>ReconcileSessionArgument</code>	<code>reconcileSessionArgument</code>	The object that specifies the servers, customer, user, and other arguments for this operation.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `reconcileSessionArgument` is null.

`OpswareInvalidArgumentException` - is thrown if the `reconcileSessionArgument.serverIDs` is null or is an empty list.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## update

Updates the attributes of the specified server. A server's attributes are used to describe:

- How the server is being used: not specified, production, and staging.
- What stage of the life cycle the server is in: decommissioned, in deployment, or live.
- How the end user wants to describe the server.

## Result

SOAP Data Type: `string`

Description: A list of all servers with updated attributes.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server whose attributes you want to update.
<code>serverUse</code>	<code>string</code>	Server use flag of server to update. If null, this argument is ignored.
<code>deploymentStage</code>	<code>string</code>	Deployment stage flag of server to update. If null, this argument is ignored.
<code>serverDescription</code>	<code>string</code>	Description of the server to update. (A short name for the description.) If null, this argument is ignored.
<code>notes</code>	<code>string</code>	Comments for the server to update. If null, this argument is ignored.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

# Chapter 13: Software Web Service

## IN THIS CHAPTER

This chapter describes the following SOAP operations:

- get
- getAssociatedList
- getIDs
- getInstalledList
- getList

The Software Web Service retrieves information on the Opware entities that represent software packages.

In Opware SAS, software packages reside in a central Software Repository. Opware SAS enables you to select software packages from the Software Repository, select the servers that you want to install the software on, preview the results of the installation, and install the software.

## get

Returns a software package object specified by ID.

### Result

SOAP Data Type: `SoftwarePackage`

Description: The object that represents the software package.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>softwarePackageID</code>	<code>decimal</code>	The ID (primary key) of the software package to be returned.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `softwarePackageID` is null.

`OpswareObjectNotFoundException` - is thrown if no `SoftwarePackage` object matches the specified `softwarePackageID`.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getAssociatedList

Returns software packages associated with a server.

### Result

SOAP Data Type: `SoftwarePackageList`

Description: An array of software package objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>decimal</code>	The ID of the server associated with a software package.

### Exceptions

`OpwareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

## getIDs

Returns an array of software package IDs that match the criteria specified by the `softwarePackageFilter` argument. For example, to search for software packages associated with one or more platforms (operating systems), for example, specify `softwarePackageFilter.platformIDs`. To search by both platforms and customers (logical AND), specify the `platformIDs` and `customerIDs` elements of the `softwarePackageFilter`. The `softwarePackageFilter.softwarePackageIDs` array must be null or empty.

```

if softwarePackageFilter is null
    throw an exception
else if softwarePackageFilter.softwarePackageIDs is non-empty
    throw an exception
else
    return an array of software package objects matching the
    conjunction of the following SoftwarePackageFilter elements:
    customerIDs, platformIDs, softwarePackageName. Null and empty
    elements are ignored.
    
```

## Result

SOAP Data Type: Decimal []

Description: An array of software package IDs.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>softwarePackageFilter</code>	<code>SoftwarePackageFilter</code>	The object that specifies the search criteria for this operation.

## Exceptions

`OpwareInvalidArgumentException` - is thrown if all the fields of `softwarePackageFilter` are null, or `softwarePackageFilter.softwarePackageIDs` is a non-empty list.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

`OpwareNullArgumentException` - is thrown if `softwarePackageFilter` is null.

`OpwareListRetrievalErrorException` - is thrown if an error occurs when retrieving the list.

## getInstalledList

Returns software packages installed on a server.

### Result

SOAP Data Type: `InstalledSoftwarePackageList`

Description: An array of software objects.

### Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>serverID</code>	<code>Decimal</code>	The ID of the server on which the software packages are installed.

### Exceptions

`OpswareNullArgumentException` - is thrown if the specified `serverID` is null.

`OpswareException` - is thrown when an application level error is encountered.

`OpswareInternalErrorException` - is thrown when an internal error is encountered.

`OpswareAccessControlException` - is thrown when the requested access is denied.

`OpswareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.



## getList

Returns an array of software package objects that match the criteria specified by the `softwarePackageFilter` argument. To search for software packages by their IDs (primary keys), specify `softwarePackageFilter.softwarePackageIDs`. To search by the other `softwarePackageFilter` elements, specify a null or empty `softwarePackageFilter.softwarePackageIDs`. To search for software packages associated with a particular customer, for example, specify `softwarePackageFilter.customerIDs`.

This operation searches for software packages with the following algorithm:

```
if softwarePackageFilter is null
  throw an exception
else if softwarePackageFilter.softwarePackageIDs is non-empty
  return corresponding array of software package objects.
else
  return an array of software package objects matching the
  conjunction of the following SoftwarePackageFilter elements:
  customerIDs, platformIDs, softwarePackageName. Null and empty
  elements are ignored.
```

## Result

SOAP Data Type: `SoftwarePackageList`

Description: An array of software package objects.

## Arguments

ARGUMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>SoftwarePackageList</code>	<code>SoftwarePackageFilter</code>	The object that specifies the search criteria for this operation.

## Exceptions

`OpwareNullArgumentException` - is thrown if the specified `softwarePackageFilter` is null.

`OpwareInvalidArgumentException` - is thrown if all the fields of `softwarePackageFilter` are null.

`OpwareException` - is thrown when an application level error is encountered.

`OpwareInternalErrorException` - is thrown when an internal error is encountered.

`OpwareAccessControlException` - is thrown when the requested access is denied.

`OpwareAuthenticationException` - is thrown when user authentication fails.

`java.rmi.RemoteException` - is thrown when a framework or network error is encountered.

# Chapter 14: Complex Data Types

## IN THIS CHAPTER

This chapter describes the complex SOAP data types used by the Opware SAS Web Services API 2.2.

## CommandSession

Provides configuration information about Command Engine sessions.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
agentName	string	Not used; to be removed in a future release.
endDate	dateTime	Date and time when the Command Engine session ended.
hostName	string	Hostname and realm of the server from where the Command Engine session was initiated.
miscDigest	string	Not used; to be removed in a future release.
myJobsVisible	boolean	Indicates if the job is displayed in the My Jobs panel of the Opware Command Center.
ownerName	string	Not used; to be removed in a future release.
paramsDigest	string	Not used; to be removed in a future release.
resultsDigest	string	Not used; to be removed in a future release.
scheduleDate	dateTime	Time and date the Command Engine session is scheduled to be run.
scheduleId	decimal	Not used; to be removed in a future release.
scriptVersionId	decimal	Version number of the Command Engine session script.
serviceInstanceId	decimal	Command Engine instance which ran (or will run) the current session.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
sessionDescription	string	The text describing the Job Name displayed in the My Jobs panel of the Opware Command Center.
sessionId	decimal	Unique identifier for each Command Engine session.
signature	string	Signature verification of the session and all auxiliary data.
signOffDate	dateTime	Not used; to be removed in a future release.
staleDate	dateTime	The point in time at which a Command Engine session will expire if the session has not been started or completed.
startDate	dateTime	Date and time when the Command Engine session started.
status	string	Status of the Command Engine session. Examples: SUCCESS, FAILED.
userId	decimal	Unique identifier of the Opware user who initiated the Command Engine session.
userName	string	Not used; to be removed in a future release; replaced by userId.

## CustomAttribute

Represents a name-value pair that can be assigned to an Opsware entity such as a customer, server, software application (node), or facility.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
key	string	The name of the custom attribute.
objectID	decimal	The unique ID of the Opsware entity the custom attribute has been assigned to. For example, if the objectType is CUSTOMER, then the objectID is the same as the customerID.
objectType	string	The type of Opsware entity the custom attribute has been assigned to. The objectType may be one of the following: CUSTOMER FACILITY NODE SERVER
value	string	The value of the custom attribute.

## CustomAttributeFilter

Determines the search results of the Custom Attribute Service `getValues`, `remove`, and `set` operations.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customAttributeKeys</code>	<code>string []</code>	An array of custom attribute names.
<code>objectIDs</code>	<code>decimal []</code>	The unique ID of the Opsware entity the custom attribute has been assigned to. For example, if the <code>objectType</code> is <code>CUSTOMER</code> , then the <code>objectID</code> is the same as the <code>customerID</code> .
<code>objectType</code>	<code>string</code>	The type of Opsware entity the custom attribute has been assigned to. The <code>objectType</code> may be one of the following: <code>CUSTOMER</code> <code>FACILITY</code> <code>NODE</code> <code>SERVER</code>

## Customer

Represents a customer entity in Opsware SAS.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
authorizationDomain	string	The authorization domain of the customer, usually the same as the domain name.
businessAccountID	decimal	TBD
customerID	decimal	The unique ID (primary key) that identifies the customer.
customerName	string	The short name of the customer.
displayName	string	The display name of the customer in the Opsware Command Center.
status	string	The status of the customer. Allowed values: ACTIVE (TBD others?)



## CustomerFilter

Determines the search results of the Customer Web Service `getIDs` and `getList` operations.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
facilityID	decimal	ID of the facility associated with customers to retrieve.
customerName	string	The name of the customer to retrieve.
customerIDs	decimal []	An array of customer IDs to retrieve.

## DistributedScriptArgument

The argument of the `execute` operation of the Distributed Script Web Service, which enables you to run and monitor a script on multiple servers.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerIDs</code>	<code>decimal []</code>	An array of customer IDs associated with the servers on which the distributed script will be executed.
<code>hiddenParameters</code>	<code>DistributedScriptParameterSet</code>	Includes the encrypted parameters such as the password.
<code>myJobsVisible</code>	<code>boolean</code>	Indicates if the distributed script execution job is visible in the My Jobs panel of the Opsware Command Center. Returns true if the job should be visible.
<code>parameters</code>	<code>DistributedScriptParameterSet</code>	The parameters required for the distributed script to be executed. For Unix scripts, use Bourne ( <code>sh</code> ) shell syntax and for Windows scripts, use <code>cmd.exe</code> parameter syntax.
<code>scheduleDate</code>	<code>dateTime</code>	The date on which to run the script. If the schedule date is not specified, the script is executed immediately.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
scriptVersion	string	The version information of the distributed script to be executed. Each time a script is modified, version information is created and stored.
sessionDescription	string	The text describing the Job Name in the My Jobs pannel in the Opware Command Center.
staleDate	dateTime	The time after which it is too late to run a scheduled distributed script. If the staleDate is later than the current date, the script will not be run.
userName	string	The name of the Opware user who executed the distributed script.

## DistributedScriptParameterSet

Contains the details for the parameter of a distributed script execution (DSE) operation on a managed server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
scriptArguments	string	The arguments of the distributed script to be executed.
scriptCodeType	string	The type of the distributed script, either Unix Shell, Windows .BAT, or Windows VBScript. Allowed values: SH BAT VBS
scriptID	decimal	The ID of the distributed script to be executed.
scriptName	string	The name of the distributed script to be executed.
scriptText	string	The contents of the distributed script to be executed.
serverDomain	string	The Windows domain of the server in which the script will be executed.
serverIDs	decimal []	An array unique IDs of the servers on which the distributed script will be executed.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
serverPassword	string	The password for the server on which the distributed script will be executed.
serverUserName	string	The user name for the server on which the distributed script will be executed.
stderrByteSize	decimal	The number of bytes in the <code>stderr</code> generated by the script.
stdoutByteSize	decimal	The number of bytes in the <code>stdout</code> generated by the script.
waitTime	int	The wait time in seconds before timing out the distributed script execution operation. This value is the amount of time a script has to complete execution on a server. If the script is not finished when the time-out value is reached, the script is stopped by Opware SAS and a script error occurs. The default value is 10 seconds.

## DSESessionCommandResult

Represents the command results of a distributed script execution (DSE) operation on a managed server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
returnCode	decimal	The return code for the command.
stderr	string	The stderr (error) text generated by the command.
stdout	string	The stdout (output) text generated by the command.

## DSESessionProgress

Represents the progress or current status of a distributed script execution (DSE) session on managed servers.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
doneServers	decimal	The number of servers that have completed the distributed script execution operation.
doneSteps	decimal	The number of completed steps for the distributed script execution operation.
errorServers	DSEServerStatus []	The servers on which the error occurred during distributed script execution operation.
hostprogress	DSEServerProgress []	An array of elements, each of which represents the distributed script execution progress of a specific server.
totalServers	decimal	The total number of servers on which the distributed script execution operation is running.
totalSteps	decimal	The total number of steps required for the distributed script execution operation.

## DSESessionResult

Represents the results of a distributed script execution (DSE) session on managed servers.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
errorServers	DSEServerStatus []	The servers on which the error occurred during distributed script execution operation.
hostsResults	DSEServerResult []	An array of elements, each of which represents the distributed script execution results of a specific server.
totalServers	decimal	The total number of servers on which the distributed script execution operation is running.



## DSEServerProgress

Represents the progress or current status of a distributed script execution (DSE) session on a specific server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
doneSteps	decimal	The number of completed steps for the distributed script execution operation.
errorDetail	string	The details of an error that might occur during distributed script execution operation.
errorName	string	The name of an error that might occur during distributed script execution operation.
serverID	decimal	The unique identifier (primary key) of the server on which the script will be executed.
stage	string	The deployment stage of the server. By default, Opsware is installed with these possible values for the deployment stage: IN DEPLOYMENT LIVE NOT SPECIFIED OFFLINE OPS READY

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
status	string	The status of the agent on the server specified by the server ID. Allowed values: OK - The server is managed by Opsware SAS NOT REACHABLE - The Server is unmanageable by Opsware SAS due to an error.
totalSteps	decimal	The total number of steps required for the distributed script execution operation.

## **DSEServerResult**

Represents the results of a distributed script execution (DSE) session on a specific server.

`DSEServerResult` extends `DSEServerStatus` and has no sub elements other than those inherited from `DSEServerStatus`.

## DSEServerStatus

Represents the status of the distributed script execution (DSE) on a specific server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
commandID	decimal	The identifier of the command running the distributed script.
errorDetail	string	The details of an error that might occur during script execution operation.
errorName	string	The name of an error that might occur during script execution operation.
serverID	decimal	The unique identifier (primary key) of the server on which the script will be executed.
status	string	The status of the agent on the server specified by the server ID. Allowed values: OK - The server is managed by Opsware SAS NOT REACHABLE - The Server is unmanageable by Opsware SAS due to an error.

## EmailNotificationSpec

Specifies the values for the “To:” and “Reply-To” in the header of the email message sent at the completion of a scheduled function (audit, snapshot, application configuration push). To specify the “From:” value, in the Opsware Command Center enter a value in the `way.notification.email.fromAddr` field. (See the *Opsware<sup>®</sup> SAS 5.2 Configuration Guide*.) The body (contents) of the email message is generated automatically.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>onSuccessOwner</code>	<code>string</code>	The email address in the “Reply-To:” of the email message sent if the function succeeds. This element may be null.
<code>onFailureRecipients</code>	<code>string []</code>	An array of email addresses of the users who receive the email message if the function fails. These addresses are in the “To:” portion of the email header.
<code>onFailureOwner</code>	<code>string</code>	The email address in the “Reply-To:” of the email sent if the function fails. This element may be null.
<code>onSuccessRecipients</code>	<code>string []</code>	An array of email addresses of the users who receive the email message if the function succeeds. These addresses are in the “To:” portion of the email header.

## EventBase

Specifies the type of event that has occurred. This type is extended by `ServerEvent`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>eventType</code>	<code>string</code>	Specifies what action has occurred. Allowed values for a <code>ServerEvent</code> : ADDED MODIFIED REMOVED

## Facility

Represents a collection of servers stored in a single Opsware Model Repository (database). A facility usually corresponds to a specific data center or geographical location (city) of an Opsware SAS installation.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
displayName	string	The name of the facility displayed by the Opsware Command Center.
facilityID	decimal	The unique identifier (primary key) of the facility.
facilityName	string	The short name of the facility.
status	string	The facility's status. Allowed values: ACTIVE DECOMMISSIONED.

## FacilityFilter

Determines the search results of the Facility Web Service `getIDs` and `getList` operations.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
customerID	decimal	ID of the customer associated with the facilities to retrieve.
facilityIDs	decimal []	An array of facility IDs to retrieve.



## InstalledSoftwarePackage

Represents a software package that has been installed on a managed server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
serverID	decimal	The unique identifier (primary key) of the server on which the software package is installed.
softwarePackageDescription	string	An informational field describing the package.
softwarePackageDisplayName	string	The name of the software package displayed in the Opware Command Center.
softwarePackageID	decimal	The unique identifier (primary key) for the SoftwarePackage object.
softwarePackageName	string	The name of the software packaged used internally by Opware SAS.
softwarePackageType	string	The type of the package, such as HOTFIX, RPM, MSI, LPP, APAR, ZIP.
softwareRelease	string	For RPMs only, the release info generated by: <code>rpm -q</code> .
softwareVersion	string	For RPM, the version info generated by: <code>rpm -q</code> . For MSI, the version specified by the user.

## ListBase

The base type for many of the <type>List complex types. For example, CustomerList and FacilityList extend ListBase. Each <type>List complex data type embeds an array of particular complex objects. For example, the CustomerList embeds an array of Customer objects.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
timeStamp	dateTime	Usually the date and time the list was retrieved.

## Node

Represents an element in the Opsware software tree.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
allowDevice	boolean	Indicates whether a server can be attached to the node.
allowMultiple	boolean	Indicates whether the stack allows a device to be attached more than once. For example, a server can only be attached to a node in the operating system stack once.
description	string	A text description of the node displayed in the Opsware Command Center.
fullName	string	The concatenation of all short names down the path to this node.
location	string	A modified version of the node's synthetic name, which specifies the full path to this node. For some stacks, prefix elements such as <code>AppServer</code> may be omitted, for example, <code>WebLogic/SunOS/5.7/8.0</code>
nodeID	decimal	The unique identifier (primary key) of the node.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
notes	string	Notes displayed in the Opsware Command Center.
parentNodeID	decimal	The identifier of the node's parent in the tree.
reconcileOrder	decimal	The default reconciliation (installation) order of the package attached to the node. This default can be overridden with install order dependencies.
shortName	string	An internal node name used by Opsware SAS. The value must be all upper case and contain no spaces.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
stackID	decimal	<p>The unique ID of a stack (subtree) in the node tree. For example, the ID of the System Utilities stack is 7. Valid IDs are:</p> <ul style="list-style-type: none"> <li>1 - customer</li> <li>2- facility</li> <li>3 - hardware</li> <li>4 - service levels</li> <li>5 - operating system</li> <li>6 - opsware</li> <li>7 - System Utilities</li> <li>8 - DatabaseServer</li> <li>9 - AppServer</li> <li>10 - WebServer</li> <li>11 - Custom Apps</li> <li>12 - OS Extras</li> <li>13 - non-existent</li> <li>14 - templates</li> <li>15 - patches</li> </ul>
status	string	<p>Has vlaues such as ACTIVE or DEPRECATED. The status is not used for software nodes.</p>
syntheticName	string	<p>The path to the node (except for the top level) displayed by the Opsware Command Center.</p>

## NodeFilter

Determines the search results of the Node Web Service `getIDs` and `getList` operations. A node is an element in Opsware's software tree.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerIDs</code>	<code>decimal []</code>	An array of customer IDs associated with the nodes to retrieve.
<code>nodeIDs</code>	<code>decimal []</code>	An array of node IDs to retrieve.
<code>nodeName</code>	<code>string</code>	The name of the node to retrieve.
<code>platformIDs</code>	<code>decimal []</code>	An array of platform IDs. A platform is the name and version of an operating system.
<code>stackIDs</code>	<code>decimal []</code>	The unique ID of a stack (subtree) in the software model tree. For a list of valid IDs, see the <code>stackID</code> description of the Node data type.

## Permission

Represents the security permission (read or write access) of an Opsware customer or facility.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
access	int	The type of access. Allowed values: 0 (none) 1 (read) 2 (read-write)
ID	string	The identifier (primary key) of the customer or facility.

## Platform

Represents the definition of an operating system in Opsware SAS. A platform is the name and version of an operating system (for example, Red Hat Linux 4.2).

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
architecture	string	The type of hardware, such as X86 or SPARC.
isPatchable	boolean	A flag is used in the Install Patch Wizard of the Opsware Command Center. If <code>isPatchable</code> is false, then the platform is not among the platforms the the user can select.
osVersion	string	Not used; to be removed in a future release.
platformDescription	string	Descriptive text or notes for the platform.
platformID	decimal	The unique identifier (primary key) of the platform.
platformName	string	The name displayed in the Opsware Command Center. For example: Red Hat Enterprise Linux AS 2.1
platformShortName	string	An internal name used by Opsware SAS. The value must be all upper case and contain no spaces.



ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
platformVersion	string	An internal name used by Opware SAS to match what the agents report with a known platform object.

## PlatformFilter

Determines the search results of the Platform Web Service `getIDs` and `getList` operations. A platform is the name and version of an operating system (for example, Red Hat Linux 4.2).

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>platformIDs</code>	<code>decimal []</code>	An array of platform IDs to retrieve.
<code>platformName</code>	<code>string</code>	The name of the platform to retrieve.

## ReconcilePackageBase

Contains the name of a the software package affected by the reconcile operation. This type is extended by ReconcilePackageDetails.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
displayName	string	The name of the software package displayed in the Opware Command Center.
uniqueName	string	The name of the software packaged used internally by Opware SAS.

## ReconcilePackageDetails

Contains information about the software package that is installed or uninstalled by a reconcile operation. `ReconcilePackageDetails` extends `ReconcilePackageBase` and is extended by `ReconcilePackageOutput`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>installFlags</code>	<code>string</code>	Optional arguments passed to the installer on the managed server.
<code>installOrder</code>	<code>decimal</code>	The default reconcile (installation) order of the package attached to the node. This default can be overridden with install order dependencies.
<code>messages</code>	<code>string []</code>	Any informational messages associated with a package that the reconcile may generate.
<code>nodeIDs</code>	<code>decimal []</code>	The identifiers of the nodes to which the software package has been attached.
<code>packageID</code>	<code>decimal</code>	The unique identifier (primary key) of the <code>SoftwarePackage</code> object.
<code>packageName</code>	<code>string</code>	The name of the package being installed or uninstalled.
<code>packageRelease</code>	<code>string</code>	For RPMs only, the release info generated by: <code>rpm -q</code>

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
packageType	string	The type of the package, such as HOTFIX, RPM, MSI, LPP, APAR, ZIP.
packageVersion	string	For RPMs, the version info generated by: <code>rpm -q</code> . For MSIs, the version specified by the user.
rebootOnInstall	boolean	Indicates whether to reboot the managed server after successful installation of the software package.
rebootOnUninstall	boolean	Indicates whether to reboot the managed server after successful uninstallation of the software package.
removeFlags	string	Optional arguments passed to the uninstaller on the managed server.
uninstallable	boolean	Specifies whether Opsware SAS has determined that the package should not be installed.

## ReconcilePackageOutput

The output returned by a reconcile operation. This type extends ReconcilePackageDetails.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
errorDetail	string	Details of any errors generated by the reconcile.
errorName	string	The name of an error generated by the reconcile.
installedDate	dateTime	The date and time when the package was installed by the reconcile.
output	string	The output generated by the tool that performed the reconcile. The tool varies with operating system. Examples of these tools are: rpm, msi, patchrm, pkgadd.
result	integer	The return code from the install or remove operation.

## ReconcileServerParameterSet

Contains the details for the parameter of a reconcile operation on a managed server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
debug	boolean	Specifies whether to run the reconcile in debug mode.
installAll	boolean	Specifies whether to install all software packages attached to the nodes to which the server has been attached.
installNodeIDs	decimal []	The identifiers of the nodes to which the software packages and server have been attached.
removeAll	boolean	Specifies whether to uninstall all software packages attached to the application nodes from which the server has been detached.
removeNodeIDs	decimal []	The identifiers of the application nodes from which software packages have been detached.
serverID	decimal	The unique identifier (primary key) of the server on which the reconcile operation will be applied.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
templateID	decimal	The identifier of the Opware template that contains the software to be installed or uninstalled.
testMode	boolean	Specifies whether this reconcile operation is a preview (test). In a preview reconcile , no software is actually installed or uninstalled.



## ReconcileServerProgress

Represents the progress or current status of a reconcile session on a specific server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
completedTasks	decimal	The number of completed steps for this reconcile operation. Steps include: downloading a package to the managed server, installing a package, and re-registering the installed package with Opware SAS.
message	string	A short text description of the current state of the reconcile operation. Example: "Querying Opware Agent for installed packages."
serverID	decimal	The unique identifier of the managed server on which the reconcile operation is running.
totalTasks	decimal	The total number of steps required for the reconcile operation.

## ReconcileServerResult

Indicates which software packages were installed or removed by the reconcile operation on a specific server. This type extends `ReconcileServerResultBase`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>installedAsSideEffect</code>	<code>ReconcilePackageBase []</code>	An array that represents the software packages installed as a side effect on the server by the reconcile operation. Examples of side effects are APARs installed when an UpdateFileset is installed, or a Solaris patch included in a Solaris package.
<code>removedAsSideEffect</code>	<code>ReconcilePackageBase []</code>	An array that represents the software packages removed as a side effect of the reconcile operation.
<code>wasInstalled</code>	<code>ReconcilePackageOutput []</code>	An array that represents the software packages successfully installed on the server by the reconcile operation.
<code>wasNotInstalled</code>	<code>ReconcilePackageOutput []</code>	An array that represents the software packages that should have been installed on the server by the reconcile operation but were not.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
wasNotRemoved	ReconcilePackageOutput []	An array that represents the software packages that should have been removed from the server by the reconcile operation but were not.
wasRemoved	ReconcilePackageOutput []	An array that represents the software packages successfully removed from the server by the reconcile operation.

## ReconcileServerResultBase

Contains information about the results of a a reconcile operation on a specific server. This type is extended by `ReconcileServerResult` and `ReconcileServerResultPreview`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>cannotBeRemoved</code>	<code>ReconcilePackageBase []</code>	An array of elements that represent the software packages that could not be removed from the server by the reconcile operation. For example, a Fileset might not be removed because all related filesets must be removed at the same time.
<code>debugError</code>	<code>string</code>	An error message generated by the reconcile operation. Appears only if <code>ReconcileServerParameterSet.debug</code> is set to true.
<code>debugWarning</code>	<code>string</code>	A warning message generated by the reconcile operation. Appears only if <code>ReconcileServerParameterSet.debug</code> is set to true
<code>error</code>	<code>string</code>	An error message generated by the reconcile operation.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
serverID	decimal	The unique identifier (primary key) of the server that was reconciled.
status	string	The status of the reconcile operation. Examples: SUCCESS, FAILURE, ABORTED, BOOTING, CYCLING.
warning	string	A warning message generated by the reconcile operation.
willNotInstall	ReconcilePackageDetails []	An array of elements that represent the software packages that Opsware SAS determined should not be installed by the reconcile operation. For example, the package may already be installed, or an UpdateFileset will not be installed because a required BaseFileset does not exist.

## ReconcileServerResultPreview

Indicates which software packages the reconcile preview (test) operation has determined would be installed or removed on a particular server. This type extends `ReconcileServerResultBase`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>willBeRemovedAsSideEffect</code>	<code>ReconcilePackageBase []</code>	An array that represents the software packages that would be removed from the server as a side effect of the reconcile operation.
<code>willInstall</code>	<code>ReconcilePackageDetails []</code>	An array that represents the software packages that would be installed on the server by the reconcile operation.
<code>willRemove</code>	<code>ReconcilePackageDetails []</code>	An array that represents the software packages that would be removed from the server by the reconcile operation.

## ReconcileSessionArgument

The argument of the `reconcile` operation of the Server Web Service, which installs or uninstalls software (applications or patches) on servers managed by Opware SAS.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerIDs</code>	<code>decimal []</code>	The identifiers of the customers associated with the servers on which the reconcile operation will be applied.
<code>globalArguments</code>	<code>ReconcileServerParameterSet</code>	The arguments common to all servers on which the reconcile operation will be applied.
<code>localArgumentsList</code>	<code>ReconcileServerParameterSet []</code>	The arguments for a single server on which the reconcile operation will be applied. A non-null element of <code>localArgumentsList</code> overrides the corresponding element in <code>globalArguments</code> .
<code>serverIDs</code>	<code>decimal []</code>	The identifiers of the servers on which the reconcile operation will be applied.
<code>userName</code>	<code>string</code>	The name of the Opware user performing the reconciliation operation.

## ReconcileSessionProgress

Represents the progress or current status of a reconcile session on managed servers.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
completedServers	decimal	The number of servers that have completed the reconcile operation.
error	string	The message of a session error.
serversProgress	ReconcileServerProgress []	An array of elements, each of which represent the reconcile progress of a specific server.
totalServers	decimal	The total number of servers on which the reconcile operation is running.



## ReconcileSessionResult

Represents the results of a reconcile operation on managed servers. This type extends `ReconcileSessionResultBase`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
serversResult	ReconcileServerResult []	An array of elements, each of which represents the results of a reconcile operation on a specific server.

## ReconcileSessionResultBase

Contains information about a reconcile operation on managed servers. This type is extended by `ReconcileSessionResult` and `ReconcileSessionResultPreview`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>completedServers</code>	<code>decimal</code>	The number of servers on which the reconcile operation has finished.
<code>error</code>	<code>string</code>	The message of an error resulting from the reconcile session.
<code>totalServers</code>	<code>decimal</code>	The total number of servers on which the reconcile operation is applied.

## ReconcileSessionResultPreview

Represents the results of a preview (test) reconcile operation on managed servers. This type extends `ReconcileSessionResultBase`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
serversResultPreview	ReconcileServerResultPreview[]	An array of elements, each of which represents the results of a preview (test) reconcile operation on a specific server

## Script

Represents a distributed script, which can be run on one or more managed servers.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
scriptID	decimal	The unique identifier of this script.
userID	decimal	The unique identifier of the Opsware user who created the script.
currentVersionID	decimal	The identifier of this version of the script. Each time a script is modified, version information is created and stored.
scriptName	string	The display name of the script.
usageNotes	string	Text that describes the script.
codeType	string	The type of the script, either Unix Shell, Windows .BAT, or Windows VBScript. Allowed values: SH BAT VBS
status	string	The status of the script. Allowed values: CURRENT DEPRECATED
createdBy	string	The name of the user who originally created the script.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
createDate	dateTime	The date and time the script was originally created.
currentVersionSetDate	dateTime	The date and time this version of the script was created.
currentVersionSetBy	string	The name of the user who created this version of the script.
sharedScript	boolean	Indicates whether or not the script is shared. If false, it is a "My Script" and can be accessed only by the user who created it.

## ScriptVersion

Represents a particular version of a distributed script. Each time a script is modified, version information is created and stored.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
scriptVersionID	decimal	The unique identifier of this version information.
scriptID	decimal	The unique identifier of the script.
version	string	The version number, for example: 30a.2.1
status	string	The status of this version. Allowed values: APPROVED ACTIVE
changeLogComments	string	Comments about the changes made to the script for this version.
uploadedBy	string	The Opsware user who uploaded this version of the script.
uploadDate	dateTime	The date and time this version was uploaded.
affectsServer	boolean	Indicates whether or not this script modifies the managed server.

## ScheduleSpec

Specifies when to perform a function (snapshot, audit, application configuration push). Because its syntax is similar to the Unix `crontab` utility, the `scheduleSpec.cronsSpec` element specifies a repeating schedule. To define a single (non-repeating) schedule, specify the `startDate` and `endDate` elements of `scheduleSpec`.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>startDate</code>	<code>dateTime</code>	The beginning date (in UTC) for this schedule. The function will not be performed until this date. If this date is null, then the schedule takes effect immediately.
<code>endDate</code>	<code>dateTime</code>	The expiration date (in UTC) for this schedule. The function will not be performed after this date. If this date is null, then the schedule does not expire. The <code>endDate</code> must be later than the <code>startDate</code> .

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
cronSpec	String	<p>The Unix <code>crontab</code> syntax that specifies the when to perform the function. For example, the following string schedules the function daily at 2:00 AM:</p> <pre>0 2 * * *</pre> <p>The next string schedules the function at 3:15 AM, Monday through Friday.</p> <pre>15 3 * * 1-5</pre> <p>For more information on the syntax, see the Unix man page for <code>crontab</code>. The <code>cronSpec</code> must not be outside of the range specified by the <code>startDate</code> and <code>endDate</code>.</p>



## Server

Represents a server machine managed by Opsware SAS.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
agentStatus	string	The status of the agent on the servers. Allowed values are: OK NOT REACHABLE
assetTag	string	The inventory asset number.
chassisID	string	A unique hardware-based identifier that the Opsware Agent discovers, typically derived from some property of the server's chassis. For this ID, Opsware SAS uses an interface's MAC address, or the hostid on Solaris servers, or the serial number for one of the interfaces.
chassisSerialNumber	string	The serial number of the hardware chassis.
defaultGateway	string	The name of the default Opsware Gateway through which the Agent on the server contacts the core.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
deploymentStage	string	The server's deployment stage. By default, Opsware is installed with these possible values for the deployment stage: IN DEPLOYMENT LIVE NOT SPECIFIED OFFLINE OPS READY
discoverDate	dateTime	The date and time the server first appeared in Opsware SAS.
effectiveBeginDate	dateTime	The date and time of the most recent hardware registration, which is performed by the Opsware agent.
hardwareSignature	string	The hardware signature used during OS provisioning on a Windows server.
lockedDate	dateTime	The date and time when the server was locked.
loopbackIP	string	The IP address for network loopback testing.
machineID	string	The MID, a unique ID of the server assigned by Opsware when the server first registers. Stored on the hard disk of the server, the MID is usually the same as the serverID.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
managementIP	string	The IP address that Opware SAS uses to communicate with the Opware Agent on the server. With static NAT, the management IP address for a server will not be the same as the primary IP address. Also with static NAT, the management IP is the NAT-translated IP address for the server. When static NAT is not being used, the management IP address is always the same as the primary IP address.
notes	string	Descriptive text for the server.
opswLifecycle	string	Specifies the life cycle phase for the server. Allowed values for a server in the server pool: AVAILABLE INSTALLING OS BUILD FAILED Allowed values for managed servers: MANAGED DEACTIVATED
origin	string	Indicates how the server got into Opware SAS. Allowed values: PROVISIONED ASSIMILATED

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
osVersion	string	The version of the operating system definition specified by the Opware administrator.
platform	Platform	The name and version of the operating system.
peerIP	string	The source IP address of the server used by the Agent during hardware registration. The peerIP is used as the default value for the management IP.
primaryIP	string	The IP address of the management interface. When you change the management interface, the primary IP changes to the IP address of that interface. The primary IP address is a locally-configured IP address.
reporting	string	The status of the agent's report of the server to Opware SAS. Allowed values: OK REGISTRATION IN PROGRESS REPORTING ERROR
serialNumber	string	The manufacturer's serial number of the server. If possible, Opware assigns the chassis ID to the serial number.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
serverDescription	string	By default, the same value as <code>systemName</code> .
serverID	decimal	A unique ID (primary key) of the managed server.
serverMfg	string	The name of the vendor that manufactured the server.
serverModel	string	The model name of the server.
serverType	string	Allowed value: <code>SERVER</code>
serverUse	string	By default, Opsware is installed with these possible values for the <code>serverUse</code> : <code>STAGING</code> <code>PRODUCTION</code> <code>NOT SPECIFIED</code> The CDR subsystem relies on <code>STAGING</code> and <code>PRODUCTION</code> .
systemDescriptor	string	Not used for servers.
systemLocation	string	Specifies the physical location of the server. For example: the facility, floor, and rack number.
systemName	string	The server name, usually the host name.
systemObjectID	string	Not used for servers.

## ServerChangeLog

Represents an entry in the change history log of a managed server. The log entries include events such as adding a server to a node, installing a template on a server, and the success or failure of a reconciliation (software installation) on a server. For a full list of the events logged, see the section “Server Histories and Reports Overview” in the *Opsware® SAS 5.2 User’s Guide*.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
changeSummary	string	A text description of the operation performed on the server.
modifiedBy	string	The name of the Opsware user who changed the server.
modifiedDate	dateTime	The date and time the change was made to the server.
serverChangeLogID	decimal	The unique ID (primary key) of the ServerChangeLog object.
serverID	decimal	The unique ID (primary key) of the changed server.

## ServerComponents

Contains a variety of attributes (primarily hardware) of a managed server.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
cpuComponent	CpuComponent []	Includes these subelements: cacheSize cpuFamily cpuModel cpuSlot cpuSpeed cpuVendor
customer	Customer	The customer associated with the server.
facility	Facility	The facility associated with the server.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
interfaceComponents	InterfaceComponent [ ]	Includes these subelements: cardIndex cardSerialNumber cpuSpeed dramBb hardwareAddress interfaceAlias interfaceDescription interfaceDuplex interfaceIndex interfaceSpeed interfaceType ipAddress localHostname memoryQuantity netMask primaryInterface slot txBandwidth
memoryComponents	MemoryComponent [ ]	Includes these subelements: memoryLocation memoryQuantity memoryType
serviceLevelNodes	Node [ ]	An array of nodes that represent user-defined categories such as Silver, Gold, and Platinum services.



ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
storageComponents	StorageComponent []	Includes these subelements: storageCapacity storageDrive storageMedia storageMfg storageModel storageType

## ServerEvent

Represents an action that has been performed on a `Server` object. The type of action is specified by the `EventBase`, which `ServerEvent` extends.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>server</code>	<code>Server</code>	The managed server on which the event took place.
<code>serverID</code>	<code>decimal</code>	The unique ID (primary key) of the <code>Server</code> object.

## ServerFilter

Determines the search results of the Server Web Service `getIDs`, `getList`, and `getEventList` operations.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>agentStatusFilters</code>	<code>string []</code>	An array of strings representing the status of the agents on the servers to retrieve. Allowed values are: OK NOT REACHABLE
<code>customerIDs</code>	<code>decimal []</code>	An array of customer IDs associated with the servers to retrieve.
<code>deploymentStageFilters</code>	<code>string []</code>	An array of strings representing the deployment stage of the servers to retrieve. By default, Opsware is installed with these possible values for the deployment stage: IN DEPLOYMENT LIVE NOT SPECIFIED OFFLINE OPS READY
<code>facilityIDs</code>	<code>decimal []</code>	An array of facility IDs associated with the servers to retrieve.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
lastRequestTimestamp	dateTime	The start date and time of the events returned by <code>getEventList</code> . The end date and time is the current date and time. Usually, <code>lastRequestTimestamp</code> is the date and time of the most recent call to <code>getEventList</code> .
machineID	string	The MID, a unique ID for the server. (For details see the <code>machineID</code> description of the <code>Server</code> type.)
serverHostName	string	The IP host name of the server to retrieve.
serverIDs	decimal []	An array of server IDs to retrieve.
serverIPAddress	string	The IP address of the server to retrieve.
serverUseFilters	string	The server "use" of the servers to retrieve. By default, Opsware is installed with these values for the server "use": STAGING PRODUCTION NOT SPECIFIED The CDR subsystem relies on STAGING and PRODUCTION.

## SoftwarePackage

Represents an installable piece of software, such as an RPM. Software packages are uploaded into the Opsware Software Repository.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
createDate	dateTime	The date and time the software package was first uploaded into the Opsware Software Repository.
createdBy	string	The Opsware user who who first uploaded the software package.
existsOnWord	boolean	Indicates whether or not the software package has been uploaded into the Opsware Software Repository.
fileSize	decimal	The size of the software package file, in bytes.
fileType	string	The type of file, such as MSI, RPM, ZIP, EXE, TAR.
installFlags	string	Optional arguments passed to the installer on the managed server.
modifiedBy	string	The name of the Opsware user who most recently modified the package properties or who re-uploaded the package.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
modifiedDate	dateTime	The date and time any property of the package is modified or when the package is re-uploaded.
notes	string	Descriptive text for the software package.
patchStatus	string	The state of the patch in its lifecycle. Allowed values: AVAILABLE DEPRECATED UNAVAILABLE UNTESTED (The UNAVAILABLE state means the patch has not been uploaded.)
rebootOnInstall	boolean	Indicates whether to reboot the managed server after successful installation of the software package.
rebootOnUninstall	boolean	Indicates whether to reboot the managed server after successful uninstallation of the software package.
removeFlags	string	Optional arguments passed to the uninstaller on the managed server.
softwarePackageDescription	string	An informational field describing the package.
softwarePackageDisplayName	string	The name of the software package displayed in the Opsware Command Center.

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
softwarePackageFileName	string	The full name of the file (not including the path) containing the package.
softwarePackageID	decimal	The unique identifier (primary key) for the SoftwarePackage object.
softwarePackageLocation	string	The fully qualified name of the file (including the path) containing the package. The file resides in the Software Repository.
softwarePackageName	string	The name of the software packaged used internally by Opware SAS.
softwarePackageType	string	The type of the package, such as HOTFIX, RPM, MSI, LPP, APAR, ZIP.
softwarePackageVersion	string	The number of times the software package has been uploaded into the Software Repository.
softwareRelease	string	For RPMs only, the release info generated by: rpm -q.
softwareVersion	string	For RPM, the version info generated by: rpm -q. For MSI, the version specified by the user.
status	string	Values include: ACTIVE DELETED

## SoftwarePackageFilter

Determines the search results of the Software Package Web Service `getIDs` and `getList` operations.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
<code>customerIDs</code>	<code>decimal []</code>	An array of IDs of the customers attached to the software packages to retrieve.
<code>platformIDs</code>	<code>decimal []</code>	An array of IDs of the platforms attached to the software packages to retrieve. A platform represents an operating system.
<code>softwarePackageIDs</code>	<code>decimal []</code>	An array of software package IDs to retrieve.
<code>softwarePackageName</code>	<code>string</code>	The name of the software package to retrieve.



## SoftwarePackageIDOverrideTuple

A tuple in an ordered list that indicates the installation order of inherited software packages. If a software package inherits other software packages in the node hierarchy, and if a server is attached to the node, then Opware SAS must determine the order in which to install these software packages.

### Elements

ELEMENT NAME	SOAP DATA TYPE	DESCRIPTION
overrideType	string	Allowed values: PLUS MINUS A PLUS indicates that the software package is inherited and installed on servers attached to this node. A MINUS means that the node is aware that its parent has a particular package, but the node does not inherit this package and servers attached to the node do not have this package installed.
softwarePackageID	decimal	The unique identifier (primary key) of the SoftwarePackage object.



# Appendix A: Compatibility With Previous Releases

## IN THIS APPENDIX

This appendix discusses the compatibility of the different versions of the Opsware SAS Web Services API.

### Running API Clients Developed for Previous Releases

The Opsware SAS Web Services API 2.2 is included with Opsware SAS 5.2. However, you can also run clients created with versions 1.0, 2.0, and 2.1 of the API with Opsware SAS 5.2. The 1.0, 2.0, and 2.1 clients access Opsware SAS through the same URL used in previous releases. To work with Opsware SAS 5.2, a 1.0 or 2.0 client that uses an Axis-generated stub needs to initialize its binding using the classes available in the Axis-generated stub.

For more information on previous releases, see “Version History” on page 4.

### Authentication and Authorization in Releases Prior to 2.1

Prior to version 2.1 of the Opsware SAS Web Services API, the API clients accessed Opsware SAS with a special user named `integration`, which resided in a different security domain than that of the Opsware Command Center. For backwards compatibility, the Opsware Installer automatically creates the `integration` user in the security domain used by the Opsware Command Center. However, if you are developing new clients or migrating 1.0 or 2.0 clients to version 2.1 (or later), you should use the Opsware Command Center to define the Opsware users and permissions needed by the clients.

For more information on security in this release, see “Authentication and Authorization” on page 17.

## Exception Handling in Releases Prior to 2.1

Before version 2.1 of the Opware SAS Web Services API, the API did not throw customized exceptions. This restriction was due to a limitation in WebLogic 7.0, which was used for the Opware Web Services Data Access Engine.

Also prior to version 2.1, Java clients had to catch a `RemoteException` and then extract the `SOAPFaultException` from the `RemoteException.detail` field. The Java clients could not catch or examine an `OpwareException` (or subclass). These restrictions do not apply 1.0 and 2.x clients that access Opware SAS.

For more information on using an `OpwareException` in this release, see “Exception Handling” on page 18.

## Changes in Operation Names From Version 1.0 to 2.0

This section is for developers who are migrating Opware SAS Web Services clients from version 1.0 to 2.0.

Except for the operation names in the Security Web Service, most of the names of the Web Services operations that are available in version 1.0 changed in version 2.0. (The Command Session and the Distributed Script Execution Web Services are new in version 2.0.) This renaming in version 2.0 rationalizes the naming conventions and allows operation overloading across Web Services.

You can still access version 1.0 operations through the 1.0 URL of the Web Server. However, an API client application can invoke *only* operations within a single version of the API. You *cannot* mix calls to version 1.0 and version 2.0 operations in the same client. Therefore, if you are calling any operations within version 1.0 of the API, all operations called from that same client application need to access version 1.0. Contact your Opware, Inc. support representative for information about continued support for version 1.0.



---

Version 1.0 clients upgrading to version 2.0 must switch from the 1.0 to the 2.0 version of the client stub. Also, the client code must be modified and recompiled.

---

The following tables map the operation names from version 1.0 to 2.0:

- Custom Attribute Web Service Operation Names

- Customer Web Service Operation Names
- Facility Web Service Operation Names
- Node Web Service Operation Names
- Platform Web Service Operation Names
- Software Web Service Operation Names
- Server Web Service Operation Names

Table 1-1: Custom Attribute Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
setCustomAttributeValue	set
getCustomAttribute	get
getCustomAttributeList	getList
getCustomAttributeKeyList	getKeyList
getCustomAttributeValues	getValues
removeCustomAttributes	remove

Table 1-2: Customer Web Service Operation Names

1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getCustomer	get
getCustomerList	getList
getCustomerIDList	getIDs
associateCustomerWithFacility	associateWithFacility
createCustomer	create
editCustomer	update
deleteCustomer	delete

Table 1-3: Facility Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getFacility	get
getFacilityList	getList
getFacilityIDList	getIDs
createFacility	create
editFacility	update
decommissionFacility	decommission

Table 1-4: Node Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
createNode	create
getNode	get
getNodeList	getList
getNodeIDList	getIDs
getParentNode	getParent
getChildrenNodeList	getChildList
getChildNode	getChild
getDescendantNode	getDescendant
attachServerToNode	attachServer
getAttachedServerList	getServerList
detachServerFromNode	detachServer
setSoftwarePackagesToNode	setSoftwarePackages
getAttachedSoftwarePackageList	getSoftwarePackageList

Table 1-4: Node Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getAttachedSoftwarePackageList-ByInheritance	getInheritedSoftwarePackageList
getSoftwarePackageID - OverrideTupleList	getSoftwarePackageIDOverrideTupleList
detachSoftwarePackageFromNode	detachSoftwarePackage
setCustomersToNode	setCustomers
getAttachedCustomerList	getCustomerList
detachCustomersFromNode	detachCustomers
setPlatformsToNode	setPlatforms
getAttachedPlatformList	getPlatformList
detachPlatformsFromNode	detachPlatforms
getNodeListForServer	getNodeList
deleteNode	delete
N/A	move (new in 2.0)

Table 1-5: Platform Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getPlatform	get
getPlatformList	getList
getPlatformIDList	getIDs

Table 1-6: Software Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getSoftwarePackage	get

Table 1-6: Software Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getSoftwarePackageList	getList
getSoftwarePackageIDList	getIDs
getInstalledSoftwarePackageList	getInstalledList
getAssociatedSoftwarePackageList	getAssociatedList

Table 1-7: Server Web Service Operation Names

VERSION 1.0 OPERATION NAME	VERSION 2.0 OPERATION NAME
getServer	get
getServerComponents	getComponents
getServerList	getList
getServerComponentsList	getComponentsList
getServerIDList	getIDs
getReachableServer	getReachableServer
getServerEventList	getEventList
getServerChangeLogList	getChangeLogList
checkServerReachability	getAgentStatus
associateServerWithCustomer	associateWithCustomer
updateServer	update
deleteServer	delete
N/A	reconcile (new in 2.0)