# Rimage Web Services MΦM Programming Guide

RIMAGE®

## Corporate Headquarters:

Rimage Corporation
7725 Washington Avenue South
Minneapolis, MN 55439
USA

800-553-8312 (toll free US)

Service: +1 952-946-0004 (Asia/Pacific, Mexico/Latin America)

Fax: +1 952-944-6956

## European Headquarters:

Rimage Europe GmbH
Albert-Einstein-Str. 26
63128 Dietzenbach Germany

Tel: +49-(0) 6074-8521-0
Fax: +49-(0) 6074-8521-100

Rimage Corporation reserves the right to make improvements to the equipment and software described in this document at any time without any prior notice. Rimage Corporation reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Rimage Corporation to notify any person or organization of such revisions or changes.

This document may contain links to web sites that were current at the time of publication, but may have moved or become inactive since. This document may contain links to sites on the Internet that are owned and operated by third parties. Rimage Corporation is not responsible for the content of any such third-party site.

# Contents

# Important Information

This section provides support contact information, cautions and warnings, and product specifications for the SDK.

## Support Information

| US, Asia/Pacific, Mexico/Latin America | Europe |
|---|---|
| **Rimage Corporation**<br>7725 Washington Avenue South<br>Minneapolis, MN 55439<br>USA<br>**Attn: Rimage Services** | **Rimage Europe GmbH**<br>Albert-Einstein-Str. 26<br>63128 Dietzenbach Germany |
| **Contact Rimage Services:**<br>**Website:** www.rimage.com/support<br>**KnowledgeBase:** http://www.rimage.custhelp.com<br>Log in and select the Ask a Question tab.<br>**Telephone:**<br>North America: 800-553-8312<br>Asia/Pacific, Mexico/ Latin America: 952-946-0004<br>**Fax:** 952-946-6956 | **Contact Rimage Services Europe:**<br>**Website:** www.rimage.de<br>**Email:** support@rimage.de<br>**Telephone:** +49-(0) 1805-7462-43<br>**Fax:** +49-(0) 6074-8521-101 |
| **When you contact Rimage Services, please provide:**<br>· System serial number and software version.<br>· Functional and technical description of the problem.<br>· Exact error message received. | **My Rimage Product Information:**<br>Copy this information from your Rimage Product for future reference.<br>**Note:** Make sure you update the Serial Number here anytime you receive a replacement autoloader.<br>**Serial Number:**<br>**Product Name:**<br>**Date of Purchase:** |

## Learn More Online

At www.rimage.com/support, you can experience Rimage's world-class Support and Services.

| From the **Support** home page: | From the product page you can access: |
|---|---|
| 1. Select your **product series**.<br>2. Select your **product**.<br>3. Learn more on the **product page**. | · Information about the latest software and firmware updates<br>· Product specifications<br>· The latest documents<br>· Current firmware and driver downloads |

# Introduction

Rimage has developed a new interface for programmers to integrate their software with the Rimage system quickly and easily. It is a web services interface which conforms to SOAP 1.1 / HTTP specification.

- Rimage Web Services are included on and installed from the Rimage Web Services SDK disc.
- Applications can consume Rimage Web Services on any platform.
- This document includes information about the Rimage Web Services, which requires Rimage Software Suite version 8.0 or above.

## Rimage Web Services Features

The Rimage Web Services interface consists of two services:

1. **RmJobService** – exposes the ability to submit jobs, cancel jobs, and obtain job statuses.
2. **RmSystemService** – exposes information about Rimage systems on the network, active alerts, and the ability to acknowledge active alerts.

Rimage Web Services eliminates the need for applications or web sites to maintain messaging connections to the Rimage systems – the web service connects automatically on the client's behalf. The client application or web site simply makes method calls to the web service to submit a job, get job status, or find out if there are any active alerts.

Also, the client doesn't have to worry about durability of jobs – all jobs are submitted durably on client's behalf. Jobs are automatically "recovered" on a web service restart.

Rimage Web Services are hosted by a Windows service on a Windows operating system. Refer to the *Installation, Configuration and Deployment* section of this document for more information.

## Getting Started

1. Install the Rimage Web Services on a Rimage embedded system, such as a Rimage 8100N or on any computer on the network.
   a. Pick the default Rimage system to use during the install.
   b. Pick the HTTP port for the Rimage Web Services to use during the install.
2. You can see the WSDL for the web services from any browser on the network. For example if the Rimage Web Service is installed on a computer named "Rimage8100N" using the HTTP port "55555", the URLs display as:
   - http://Rimage8100N:55555/RmJobService.svc?wsdl
   - http://Rimage8100N:55555/RmSystemService.svc?wsdl
3. Once Rimage Web Services are installed and started they are ready to be consumed.
   a. Rimage Web Services installation CD includes client side proxy source files for C# and VB.NET languages. These files can be used directly in the client application or web site project to use the Rimage Web Services.
   b. Installation and the CD include a *samples* folder, which contains samples for C#, VB.NET, Java, PHP, and Ruby languages.

c. For other languages any number of web service consuming technologies is available. Typically a command line utility is run against a web service URL, http://Rimage8100N:55555/RmJobService.svc or http://Rimage8100N:55555/RmSystemService.svc in the above example. These utilities generate the client proxy code in the target language so a client application can consume the web service. Some languages, such as PHP, do not require client side proxy code to be generated and can consume web services directly.

# Working with Rimage Web Services

Rimage Web Services methods follow a request / response calling convention. Every method requires a request object that is specific to that method. Every method returns a response object specific to that method. For example, a **RmJobService.SubmitUdfJob** method takes in a **SubmitUdfJobRequest** object and returns **SubmitUdfJobResponse** object.

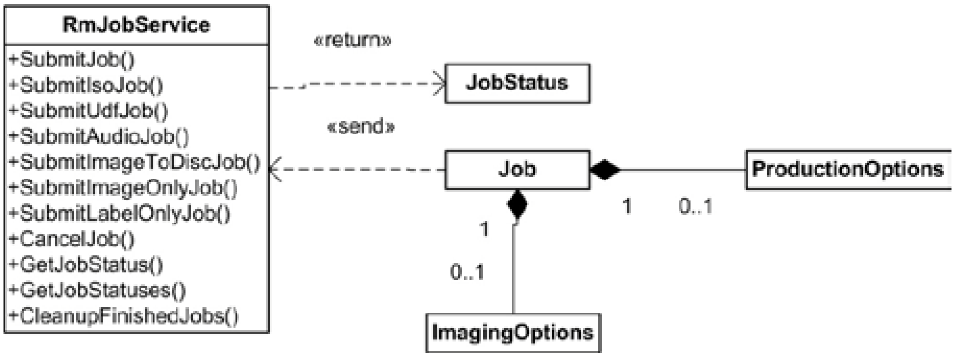
The request object contains all the input parameters, as properties, specific to a method. In the above example, the **SubmitUdfJobRequest** contains a list of files or folders to record onto the disc, UDF format options, optional label and merge file information, etc.

The response object contains the return values for the method including any error information in case the method call fails. In the above example, the **SubmitUdfJobResponse** object contains a *JobStatus* object, which has information about the immediate state of the submitted job and other pertinent information. In case of a method failure the *ErrorMessage,* and a possible *ErrorCode,* properties will contain information about why the call failed.

- **Notes:**
  - Every method requires a CallerId parameter. The CallerId value should be different for each client application or web site consuming the Rimage Web Services. In most cases the CallerId is simply logged and is used for troubleshooting in case of problems. In some methods, such as RmJobService.CancelJob, it is used to restrict an operation to a specific client.
  - No sequence is imposed on method calls within a service or between services. In other words the method calls can be mixed and matched in any desired combination

## Working with the Job Service

**RmJobService** allows the caller to submit and cancel jobs and to retrieve job statuses. The UML diagram below lists the **RmJobService** methods as well as referenced **Job** and **JobStatus** objects.



### Job Submission

Job submission methods are broken into two groups: Specific job submission methods (such as **SubmitAudioJob**) and a generic job submission method (**SubmitJob**).

Specific job submission methods cover the most common Rimage job types as well as the most common input data used with those job types. The specific methods are described below.

- **SubmitIsoJob** - Method for creating ISO discs on any kind of media (CDR, DVDR, or BDR). This method requires a list of files or folders to record, ISO level information, number of discs to make, optional JobId, and optional label and merge files.

- **SubmitUdfJob -** Method for creating UDF discs on any kind of media (CDR, DVDR, or BDR). This method requires a list of files or folders to record, UDF version information, number of discs to make, optional JobId, and optional label and merge files.
- **SubmitAudioJob** - Method for creating Audio CDs. This method requires a list of audio files to record, number of discs to make, optional JobId, and optional label and merge files.
- **SubmitImageToDiscJob -** Method for creating discs from an existing image file on any kind of media (CDR, DVDR, or BDR). This method requires the path to the image file to record, number of discs to make, optional JobId, and optional label and merge files.
- **SubmitImageOnlyJob -** Method for creating an image file on the hard drive from the specified data folders and files. This method requires a list of files or folders, ISO or UDF formation information, and an optional JobId.
- **SubmitLabelOnlyJob -** Method for printing a label on a blank or a pre-recorded disc. This method requires number of discs to make, optional JobId, label file, and an optional merge file.

Generic job submission methods include only one – **SubmitJob**. This method allows for more control of disc creation and is used in small percentage of cases where the specific job submission methods do not allow finer grained control:

- **SubmitJob** - Method for creating data discs on any kind of media (CDR, DVDR, or BDR). This method requires just one object – **Job**, as shown in the diagram above. The **Job** object describes how the data is to be imaged and recorded onto the disc and what label is to be printed on the disc. The **Job** object allows separate **ProductionOptions** and **ImagingOptions** objects to be specified. Therefore many more settings and options are available on the **Job** object than are available on the specific job submission methods.

- **Notes**:
  - **JobId** parameter is always optional and if not specified is generated by the **RmJobService**.
  - It is required that every parameter is set explicitly in the request object and other objects referenced by the request object, such as **Job**. Defaults are not available. A good example of this is PerfectPrintAngle parameter. In most cases Perfect Print feature is not used, in which case this parameter needs to be set by the client to -1.0f.

## Merge Field Information

There are three ways to specify merge field information – merge file, merge fields, or merge lines.

Merge file specifies a file on disk / network which has to be accessible to the Web Services process. For more information about the format of merge files refer to the *Using Label Merge Fields* guide.

Merge fields are specified using the MergeData FieldValues two dimensional array. The first dimension of the array specifies number of copies – number of discs produced. (each disc copy can have unique merge field values). The first dimension of the array corresponds directly to the number of lines in a merge file. The second dimension of the array specifies the actual merge field values.

In some cases it is not possible to use the two dimensional arrays with SOAP (some SOAP clients / languages don’t support this feature). In this case a single dimensional array is available in MergeData MergeLines array. Each string in this array corresponds directly to a line in a merge file. A sample string would look like this:

```
"\"Value1\",\"Value2\",\"Value3\",\"Value4\",\"Value5\",\"Value6\"
```

## Job Cancellation

- **RmJobService** offers the ability to cancel an in process job:
- **CancelJob –** *JobId* to identify a job to cancel is required in the **CancelJobRequest** object. The *CallerId* in the **CancelJobRequest** object is checked against the *CallerId* tracked with the job to make sure that the cancel request came from the "submitting" application.
- **Notes:**
  - Applications are only allowed to cancel their own jobs.
  - Jobs can be canceled whether they are in a pending or in process state, as long as they have been submitted.

## Job Status

**RmJobService** offers the ability to retrieve job status for a single job as well as for all jobs that the service knows about. Specific methods are described below.

- **GetJobStatus** - Method for retrieving a job status for a specific job. The **GetJobStatusRequest** object, passed into this method, requires a *JobId* to identify a specific job. If a job for that *JobId* isn't found, error information is returned in the **GetJobStatusResponse** object**.** If a job for the *JobId* is found, then a **JobStatus** object is returned in the **GetJobStatusResponse** object.
- **GetJobStatuses –** Method for retrieving a list of job statuses. The **GetJobStatusesRequest** object, passed into this method, requires a status filter to filter the returned list on job state (Active, Failed, etc.). *CallerOnly* Boolean parameter restricts the returned list to statuses for jobs submitted by this calling application or by all applications.
- **Note**: If no job statuses are found to match the filter criteria, the returned list is empty.

## Job Cleanup

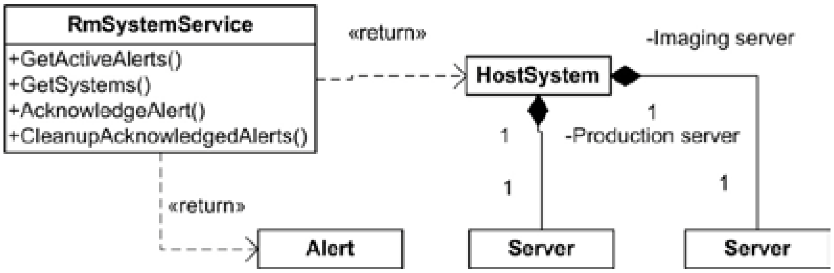
The final method of the **RmJobService** allows the caller to clean up jobs that have been completed:

**CleanupFinishedJobs** - Method for removing from the web service memory jobs that have been either completed successfully, failed, or canceled.

- **Note**: This method does not differentiate between different calling applications. It removes all finished jobs.

# Working with the System Service

**RmSystemService** allows the caller to get information about Rimage servers on the network and currently active server alerts. The UML diagram below lists the **RmSystemService** methods as well as **Server** and **Alert** properties.



## RimageSystems

**RmSystemService** offers the ability to retrieve Rimage systems on the network:

**GetSystems** - Method for retrieving a list of currently available Rimage systems. The **GetSystemsResponse** object, returned from this method, contains a list of **HostSystem** objects. **HostSystem** object contains references to any Imaging and Production servers on the system. If Production server is running, the **HostSystem** will then have information about the **Autoloader**, **Printer**, **Recorders**, and **Bins**.

- **Note**: If no Rimage systems are found on the network, the returned list is empty.

## Alerts

**RmSystemService** offers the ability to retrieve currently active server alerts as well as the ability to answer, or acknowledge, those alerts:

- **GetActiveAlerts** - Method for retrieving a list of currently active server alerts. The **GetActiveAlertsResponse** object, returned from this method, will contain a list of active **Alert** objects, if any. An **Alert** object contains information about the server which published the alert, a description of the alert, as well as possible choices for acknowledging the alert. *AlertId* identifies the alert. The **Alert** object also includes separate lists of *ReplyStrings* and *ActionIds*. Both of these lists are correlated so that the index of ReplyString is the same as the index of the corresponding ActionId in the other list.
    - **Note**: If there are currently no active alerts, the returned list is empty.

Ÿ **AcknowledgeAlert** - Method for answering an active alert. The **AcknowledgeAlertRequest** object, passed into this method, requires alert identification as well as the information about how to acknowledge the alert. A server alert is uniquely identified by a combination of *ServerId* and *AlertId*. The *ActionId* handles the acknowledging part and is one of the *ActionIds* that is listed in the **Alert** object itself.

- **Note**: The list for all possible Action IDs can be found in the *Server Alert Action IDs* reference document.

## Alert Cleanup

The final method of the **RmSystemService** allows the caller to cleanup alerts that have been acknowledged:

**CleanupAcknowledgeAlerts** - Method for removing from the web service memory alerts that have been acknowledged are no longer active.

# Error Codes

In case of failure, Rimage web services always return an error code and an error message in the Response object. This section lists all the error codes and corresponding messages. Because Rimage web services relies on the Rimage Rapid API for communication with Rimage systems, the Rapid API error codes are listed first, followed by web services specific error codes.

| Rapid API Error Codes | |
|---|---|
| 10000 | Connection with the messaging server has not been established. |
| 10001 | Connection for a certain connection id is not found. |
| 10002 | Problem creating XML Parser, no further information available. |
| 10003 | Problem parsing XML document, certain error code returned. |
| 10004 | Unrecognizable XML document type from the server, typically related to order status. |
| 10005 | Index is outside of existing range. |
| 10006 | Problem parsing XML document, no further information available. |
| 10007 | Cannot create XML writer - unrecognized server type. |
| 10008 | XML string is null. |
| 10009 | XML server reply came back empty. |
| 10010 | Problem disconnecting from a certain messaging server. |
| 10011 | Iterator type does not match the type of requested object. |
| 10012 | Server order is either Pending or Completed - cannot change. |
| 10013 | Cannot open log file for writing. |
| 10014 | Level is not specified for certain format. |
| 10015 | Problem getting log file path from .\\rapid_log.properties file. |
| 10016 | Log file Append setting isn't specified in the rmapi_log.properties file. |
| 10017 | Problem with a certain setting in the rmsclrt_log.properties file. |
| 10018 | Quantity of discs cannot be outside of 1 to 30000 range. |
| 10019 | Server for a certain server Id was not found. |
| 10020 | Server is password protected, call VerifyPassword first. |
| 10021 | Passed in value cannot be null. |
| 10022 | Specified media type is invalid. |
| 10023 | Specified media size is invalid. |
| 10024 | Bin usage value is invalid. |
| 10025 | Mailslot usage value is invalid. |

| | |
|---|---|
| 10026 | Production server mode value is invalid. |
| 10027 | Production operating mode value is invalid. |
| 10028 | Maximum recording speed value is invalid. |
| 10029 | Add postgap to ISO value is invalid. |
| 10030 | Printing reject pattern value is invalid. |
| 10031 | Reading preferred reader value is invalid. |
| 10032 | Reading audio read speed value is invalid. |
| 10033 | Auto bulk read audio track value is invalid. |
| 10034 | Production kiosk mode value is invalid. |
| 10035 | Reply text doesn't match a valid reply choice. |
| 10036 | A certain parameter cannot be null. |
| 10037 | A certain object cannot be null. |
| 10038 | Specified job is already submitted or in process. |
| 10039 | Specified job is not an ImageOnlyJob. |
| 10040 | Specified job is not an ImageAndRecordJob. |
| 10041 | Specified job is not a ProductionOnlyJob. |
| 10042 | Failed in creating XML string for certain job and type. |
| 10043 | Must specify a host name to start a server. |
| 10044 | Calling process does not have a valid UI Window. |
| 10045 | Thread ID is invalid in calling process. |
| 10046 | No object is created. |
| 10047 | Specified job type is out of type range. |
| 10048 | Invalid value for Image File Type. |
| 10049 | Specified is not a valid boolean value. |
| 10050 | Invalid value for Image File Size. |
| 10051 | Invalid value for UDF Version. |
| 10052 | Invalid value for UDF Apple Extension. |
| 10053 | Invalid value for ISO Level. |
| 10054 | Specified job is missing image format. |
| 10055 | Can't set ISO params when UDF format is set. |
| 10056 | Specified job should have at least one Parent folder or one EditList. |
| 10057 | Media type for a job is invalid. |
| 10058 | Media size for a job is invalid |

| | |
|---|---|
| 10059 | Passed in parameter can't be empty. |
| 10060 | File name is too short. |
| 10061 | File can't end with a slash. |
| 10062 | Specified file parameter requires a full path. |
| 10063 | Specified file does not exist. |
| 10064 | Invalid label file path. |
| 10065 | No parameter was set for a job. |
| 10066 | A job has no Data Tracks nor Audio Tracks. |
| 10067 | Media type is not set on a job. |
| 10068 | Image file size is too large for Media type. |
| 10069 | Job needs None set for Iso level. |
| 10070 | Job needs None set for Iso extension. |
| 10071 | Job needs Iso level set. |
| 10072 | Job needs Udf version set. |
| 10073 | Job needs Hfs Iso Apple extension set. |
| 10074 | Job does not contain Audio tracks. |
| 10075 | Job does not contain Data tracks. |
| 10076 | Job is missing image file path. |
| 10077 | Specified job id already exists. |
| 10078 | Invalid priority specified. |
| 10079 | User job parameter ID is invalid, it must be outside of reserved range. |
| 10080 | Job is missing label file path. |
| 10081 | Specified label file does not exist. |
| 10082 | Specified Dtd file path is invalid; it does not contain \\XML. |
| 10083 | Specified Dtd file path is invalid, it does not contain Dtd name. |
| 10084 | Bad job cast. |
| 10085 | Xml header does not contain "<!DOCTYPE". |
| 10086 | Invalid XML header, the string is empty. |
| 10087 | Xml header is invalid, it does not cotain <!. |
| 10088 | Buffer size is too small. |
| 10089 | Job id has to be specified. |
| 10090 | Specified Job is not a valid subclass of the specified interface. |
| 10091 | Error in converting specified path to UNC format. |

| | |
|---|---|
| 10092 | Invalid UNC path. |
| 10093 | Passed in number param is not a valid numerical string. |
| 10094 | Number of copies can't be 0 or a negative number. |
| 10095 | Label Only Job cannot have Audio or Data tracks. |
| 10096 | Thread ID 0 is invalid in the calling process; If you do NOT want the Listener to be called in main UI thread, use UIFalse instead. |
| 10097 | Thread ID 1 is invalid in the calling process; If you WANT the Listener to be called in main UI thread, use UITrue instead. |
| 10098 | Image file size is too large for media type. |
| 10099 | Incorrect parameter for Label type. |
| 10100 | Cannot set Udf format when Iso is specified. |
| 10101 | Specified job type is invalid. |
| 10102 | Joliet needs ISO Level set to 2. |
| 10103 | ISOAppleExt must be set to HFS. |
| 10104 | ISO level must be set to "none". |
| 10105 | Apple Ext can't be HFS. |
| 10106 | Apple Ext must be set to UDF102. |
| 10107 | Both ISO Level and ISO Apple Ext are set to "none". |
| 10108 | Enum value is out range. |
| 10109 | Job Type is out of range. |
| 10110 | Job type is not a valid job type, none of the Job Factories recognize it. |
| 10111 | Rapid is already connected to a messaging server. |
| 10112 | Server is offline. |
| 10113 | ImageOnly jobs cannot accept email address parameter. |
| 10114 | Cannot have Data tracks in Audio type job. |
| 10115 | Specified job was not created by RAPID. |
| 10116 | Job type is not set yet. |
| 10117 | Invalid job type. |
| 10118 | Can't reset job type on a job, when its type is already set. |
| 10119 | Job for specified job id doesn't exist. |
| 10120 | Failed to allocate memory. |
| 10121 | Failed to allocate memory. |
| 10122 | Can't change job ID when a job is sumbitted or in process. |
| 10123 | The iterated object is null. |

| | |
|---|---|
| 10124 | The iterated object is invalid. |
| 10125 | Can't set CDXA to true for Hybrid job. |
| 10126 | Can’t have certain characters in xml. |
| 10127 | Xml writer needs a valid Job object. |
| 10128 | Server parameter constant isn't in the ServerConsts table. |
| 10129 | The specified directory for durable jobs is invalid. |
| 10130 | Invalid number of recording retries. |
| 10131 | ReadJob requires IReadJobStatusListener |
| 10132 | Disc map file is invalid. |
| 10133 | Disc map file does not exist. |
| 10134 | Error in reading disc map file. |
| 10135 | Error parsing disc map file. |
| 10136 | Disc track not found. |
| 10137 | Specified job is not an object of ReadJob or its subclass. |
| 10138 | Can't add read action after the read has ended. |
| 10139 | ReadSpeed is invalid for CopyJob. |
| 10140 | Job %1: value of parameter %2!d! (%3) can't be empty. |
| 10141 | User doesn’t have permission to write to a folder. |
| 10142 | Invalid value for PerfectPrintAngleParam, it must be multiple of 0.25 between 0 and 359.75. |
| 10143 | Hybrid disc cannot have the sector size changed. |
| 10144 | Unknown exception thrown in onStatus() call. |
| 10145 | Job ID cannot be null or empty. |
| 10146 | Failed opening original merge file. |
| 10148 | Failed opening temporary merge file. |
| 10149 | Original merge file is empty. |

| Web Services Error Codes | |
|---|---|
| 50000 | General error. |
| 50001 | Request object itself is null. |
| 50002 | ImagingOptions in the Job object is not specified. |
| 50003 | ProductionOptions in the Job object is not specified. |
| 50004 | Number of copies is less than 1. |
| 50005 | File object in the Job object is not specified. |
| 50006 | List of files to image is empty. |
| 50007 | Dvd Protect option doesn’t apply to the specified format. |
| 50008 | Image file path is not specified. |
| 50009 | Label file path is not specified. |
| 50010 | JobId is not specified. |
| 50011 | Requested job is not found. |
| 50012 | AlertId is not specified. |
| 50013 | ServerId is not specified. |
| 50014 | ActionId is not specified. |
| 50015 | Unknown Job type. |
| 50016 | Merge data is not specified. |
| 50017 | CallerId did not match a required CallerId. |
| 50018 | Job object is not specified. |

# Installation, Configuration and Deployment

## Installation

Rimage Web Services SDK is distributed on a CD which contains an install program. The installation is performed on a server machine that will host the Rimage Web Services. This may be a Rimage embedded system, such as an 8100N, a Rimage control center system, or a completely separate machine.

The installation presents several choices, such as where the Rimage Messaging server is running and whether the Rimage Web Service should be exposed through an HTTP port number different from the default. The default is port 55555.

Rimage Web Service is hosted by a Windows service, installed under the name **Rimage_WS**.

If the hosting computer is part of a Windows domain, the Rimage_WS service is installed using the **NT Authority\networkservice** account. This account should have access to network shares on other computers that are also part of the same domain.

If the hosting computer is not part of a domain, the Rimage_WS service is installed using the **NT Authority\localservice** account. This account only has access to the local computer. If access to network shares is required, then this account needs to be changed to some other account which also exists on network computers.

Supported Windows platforms are:

- Windows XP (excluding Windows XP Home)
- Windows Vista (excluding Vista Home Basic and Vista Home Ultimate)
- Windows 7 (excluding Windows 7 Starter and Home Premium)
- Windows Server 2003
- Windows Server 2008

The runtime files, PDF manuals, and HTML help files are located in the installation folder under the **doc** subfolder.

Rimage Web Services make a connection to the Messaging server specified during the install and maintain this connection for the lifetime of the web service.

## Configuration

The Rimage Web Service is externally configured through a *web.config* file located in the installation folder. This file describes the bindings and endpoints for the **RmJobService** and **RmSystemService** services. These sections are typically not altered by the user.

The **web.config** file also contains application level settings in the <*appSettings*> section:

- *MessagingHost* – specifies the hostname or the IP address of the Messaging server for the web services to connect to. This value is set during the install.
- *MessagingPort* - specifies the port to use for the Messaging server connection. This value is set during the install.
- *RimageSystemFolder* – specifies the path to the Rimage System Folder (RSF) which contains subfolders for log files, DTD files, etc. This value is set during the install.
- *MaxLogFileSize* – specifies the maximum size that the web services log file can grow to. After this size is reached the log file is recycled. The default value is 2 MB. The log file is located in <*RSF*>*logs* folder and is typically named <*hostname*>*_RmWebService.log*.
- *ConvertToUNC* – specifies whether the file and folder paths used during job submission are converted to UNC, if possible, or used as is. The default value is **true**.

# Referenced Documents

- Information about Server Alert Action IDs can be found in the *Server Alert Action ID Reference* document.
- Information about Editlists, can be found in the *Using Edit Lists* guide.
- Information about label merge fields can be found in the *Using Label Merge Fields* guide.
- Information about server alert Action IDs can be found in the *Using CD Text* document.