

Saxonite Limited

ArtiSan Mailbox Manager

AMMDiagnostics

Author: Keith Young
Reference: ArtiSan/Rel1.1
Date: 26 April 2006
Version: 1.0
Distribution: Company confidential

Abstract

This document the AMM diagnostics tool for ArtiSan

© Saxonite Ltd 2006

Change History

1.0 Initial

Contents

Notices	2
ArtiSan Diagnostics	3
Web URL access	3
Archive Engine access	4
Database and Storage Driver access	5

Notices

Copyright © 2005 Saxonite Limited. All rights reserved.

No part of this book, including its interior design, and icons, may be reproduced, stored in a retrieval system, or transmitted in any form by any means, electronic, mechanical, photocopying, recording or otherwise, without the express written permission of Saxonite Limited.

LIMITED LIABILITY; DISCLAIMER OF WARRANTY

SAXONITE LIMITED MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS DOCUMENT AND SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY IS TO BE MADE EITHER BY SALES REPRESENTATIVES OR THROUGH ANY WRITTEN SALES MATERIALS. SAXONITE LIMITED SHALL NOT BE LIABLE FOR ANY LOSS OF PROFITS OR ANY OTHER COMMERCIAL DAMAGES, INCLUDING BUT NOT LIMITED TO SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR OTHER DAMAGES.

The publisher of this document is

Saxonite Limited.

Unit 101, Spitfire Studios,

63-71 Collier St,

London, N1 9BE

Internet: www.mailboxmanagement.com

Windows 2000/2003 Server and Internet Explorer are trademarks of Microsoft.

All other trademarks, trade names, or product names used in this document are the property of their respective owners.

© 2005 Saxonite Limited. All rights reserved.

AmmDiagnostics is intended for Administrators of AMM.

AMMDiagnostics runs 3 types of tests:

1. Web URL access
2. Archive Engine access
3. Database and Storage Driver access

If one full test succeeds, it results in retrieving the message from the archive and enabling the administrator to forward the message as an attachment to any email address. If the test succeeds, no further tests are performed.

If a test fails, the error is shown, and the next test starts.

Using this tool, if retrieving the message via the Web URL works first time, and a user has reported a problem, then this proves the problem is specific to that user. You need to check network and/or internet connectivity as applicable for that user to the Web URL he provided.

If the Web URL access fails, it will show an error. You need to investigate this error in order to fix the problem.

The following tests are made to retrieve the message and allow for it to be forwarded to the end user to help in minimising inconvenience when a problem occur and in cases when retrieving the archived message is of utmost importance.

It is important to understand that layer in this architecture relies on the layer underneath. If there is a problem with the Archive Engine, then it is very likely the Web URL access will fail too. If the storage device is down, all tests will fail.

When investigating a problem, always start with the last error message in the log, and then test again.

Web URL access

1) `m_spRequest.CreateInstance(_T("MSXML2.XMLHTTP"))`

Failure : Failed to create the msxml object. Is msxml installed? Check system32 for msxml.dll.

Success : Created MSXML object.

2) `m_spRequest->raw_open((_bstr_t)_T("GET"), (_bstr_t)strURLCopy, (_variant_t)VARIANT_FALSE);`

Failure : Failed initialising the request (open). Check the URL is valid.

Success : Initialised the request (open).

3) `m_spRequest->raw_send();`

Failure : Failed sending the request (send). Check the URL is valid, check your network and/or internet connection (as required) works. Is the target server up and running? is it responding?

Success : Sent the request (send).

4) `m_spRequest->get_readyState(&ready_state);`

Failure : Failed to get ready state. You should not see this message, something's gone wrong with the http request.

Success : Got ready state.

5) `if(ready_state < 4)`

Failure : The URL request is uninitialised (0), loading(1), loaded (2) or interactive (3), but not ready. Its ready state is < ready_state >. You shouldn't see this message as the request is synchronous.

Success : Ready state ok. (>=4)

6) `m_spRequest->get_status(&status);`

Failure : Failed to get HTTP status code. It seems the request hasn't returned a response.

Success : Got HTTP status code < status >.

7) `if(status > 299)`

Failure : Failed with HTTP status code: < status >, < status_text >. This is a standard code returned by the target web server, the problem therefore needs to be investigated on the web server side.

8) `m_spRequest->get_responseText(&bstr_text);`

Failure : Failed to get response body. The target web server has returned a response, but we can't get the response body.

Success : Got response body.

9) `spMessage.CreateInstance(_T("CDO.Message`

Failure : Failed to create CDO Message while testing the Web URL. CDO should be installed as part of the Operating System (cdosys.dll for example) but also comes with CDONTS, Exchange server...

Success : Created CDO Message after testing Web URL.

10) `spMessage->raw_GetStream(&spStream);`

Failure : Failed to get stream for the message. AMM Diagnostics Internal error.

Success : Got stream for the message.

11) `spStream->raw_WriteText(bstr_text,ADODB::adWriteChar);`

Failure : Failed to write response text into message stream. AMM Diagnostics Internal error.

Success : Written response text into message stream.

12) `spStream->raw_Flush();`

Failure : Failed to flush data to the message stream. AMM Diagnostics Internal error.

Success : Flushed data to the message stream.

Archive Engine access

1) `url.Split(_T(sURL))`

Failure : Failed to parse the URL. Check the asp page name is followed by a ? and name=value pairs separated by & with a = sign.

Success : URL Parsed.

2) `url.GetQueryStringValueForKey("id",&strID)`

Failure : Failed to get the ID from the parsed URL. Check the URL, does it have an id followed by a value? E.g. `http://server/page.asp?Id=ABCDEFGH`

Success : ID extracted from parsed URL.

3) `spEngine.CreateInstance(_T("ArtiSan.ArchiveEngine"))`

Failure : Failed to create the ArtisanMM.Engine object. Is the Artisan software installed? Check Engine.dll in the installed software Bin directory. It needs to be registered using regsvr32.exe.

Success : ArtisanMM.Engine object created.

4) `spEngine->RetrieveMessage(SXN::AutoBSTR(_T(strID)), &spMessage)`

Failure : Failed to retrieve the message using the ArtisanMM.Engine object. The message Id might be incorrect or does not exist in the archive or another problem occurred.

Success : Message retrieved using ArtisanMM.Engine object.

Database and Storage Driver access

1) `url.Split(_T(sURL))`

Failure : Failed to parse the URL. Check the asp page name is followed by a ? and name=value pairs separated by & with a = sign.

Success : URL Parsed.

2) `url.GetQueryStringValueForKey("id",&strID)`

Failure : Failed to get the ID from the parsed URL. Check the URL, does it have an id followed by a value? E.g. http://server/page.asp?Id=ABCDEFGH

Success : ID extracted from parsed URL.

The following section deals specifically with DB connectivity (to retrieve driver and storage id)

3) `GetConfigString(ARTISAN_DSN, DEFAULT_ARTISAN_DSN);`

Failure : No DSN name defined in Artisan configuration. Use ArtiSanManager web site to configure the DSN.

Success : DB DSN retrieved from Artisan configuration.

4) `GetConfigString(ARTISAN_DBUSER, DEFAULT_ARTISAN_DBUSER);`

Failure : No DB username defined in Artisan configuration. Use ArtiSanManager web site to configure the DSN.

Success : DB username retrieved from Artisan configuration.

5) `GetEncryptedConfigString(ARTISAN_DBPASSWORD, DEFAULT_ARTISAN_DBPASSWORD);`

Failure : No DB password defined in Artisan configuration (check DSN).

Success : DB password retrieved from Artisan configuration.

6) `m_Connection.Open(dsn,user_name,password);`

Failure : Failed to open connection to DB using details from DSN. Check the DSN details are correct on the ArtiSanManager web site. Also check the System DSN tab in Windows Data Source/ODBC settings, and make sure Test Connection is successful. Check the DB server is up.

Success : DB connection opened.

7) `m_Connection.Query(cmd, spRS);`

Cmd is as follow:

```
SELECT StoreDriverId,StorageId FROM ArtiSan where AMMid=' strID '
```

Failure : Failed to submit query to DB. The DB query wasn't parsed successfully by the DB server. Are there invalid characters in the URL? A full DB error message should precede this line.

Success : Submitted query to DB.

8) `m_Connection.GetField(spRS, _variant_t(_T("StoreDriverId")), var_value);`

And also:

```
m_Connection.GetField(spRS, _variant_t(_T("StorageId")), var_value2);
```

Failure : Failed to get StoreDriver and Storage IDs fields from DB table. Retrieving values for fields StoreDriverId and/or StorageId returned an error. Check the DB table design.

9) rows affected: 0

Failure : No record found for this message ID in DB table. This message Id doesn't seem to exist in the DB. Has this message been archived? Does the DB table contain any rows at all?

10) rows affected: >1

Failure : This message id exists more than once in the DB, this is not expected. You should never see this because of a DB constrain. Has the DB design been changed?

Success : StoreDriver and Storage IDs retrieved from DB table.

The following section deals with retrieving the actual data from the storage (code mainly copied/pasted from Archiver).

Section 11 deals with a specific driver, and section 12 with the default driver.

--- start of choice of path 11 or 12---

11a) Look for key

```
key = REGISTRY_KEY\StorageDrivers\
```

Success: Found StorageDriver key in registry.

11b) `prog_id = key.GetString(_T(""))`

Failure : Invalid storage drivers configuration: empty ProgId. The registry key for StorageDrivers exists, but hasn't got expected values.

Success : Retrieved progID from registry.

11c) `spSD.CreateInstance(prog_id);`

Failure : Failed to create Storage Driver instance from progID. This could be due to a bad installation of the Storage Driver. Maybe try to re-install? (progid=<value>)

Success : Created storage driver instance from progID.

11d) `spSD->Init(_bstr_t(reg_area));`

```
reg_area = _T("\\StorageDrivers\\") + strStoreDriverId;
```

Failure : Failed to initialise storage driver. Instance was created successfully, but failed to initialise. (progid=<value>, StoreDriverID=<value>)

Success : Initialised storage driver.

12a) Using default

Success: No storage drivers configured in registry for archiving data, using default.

12b) spSD.CreateInstance(_T("ArtisanMM.FileStorage"))

Failure : Failed to create default storage driver instance (ArtisanMM.FileStorage object).
Is the Artisan software installed? Check archiver.dll typically in installed Bin directory. It needs to be registered using regsvr32.exe.

Success : Default StorageDriver instance created.

12c) spSD->Init(_bstr_t(_T("")));

Failure : Failed to initialise the default storage driver. Instance was created successfully, but failed to initialise.

Success : Initialised the default storage driver.

--- end of choice of path 11 or 12---

13) spSD->LoadFromStore(_bstr_t(strStorageId), &spMI, &var_stream);

Failure : Failed to load message from store using storage driver. Is the storage unit up and running?

Success : Message loaded from store using storage driver.

14) spMessage.CreateInstance(_T("CDO.Message

Failure : Failed to create CDO Message while testing DB and StorageDriver. CDO should be installed as part of the Operating System (cdosys.dll for example) but also comes with CDONTS, Exchange server...

Success : Created CDO Message while testing DB and StorageDriver.

15) spMessage->raw_GetStream(&spStream);

Failure : Failed to get stream for the message. AMM Diagnostics Internal error.

Success : Got stream for the message.

16) spStream->put_Type(ADODB::adTypeBinary);

Failure : Failed to make message stream binary. AMM Diagnostics Internal error.

Success : Changed message stream to binary type.

17) spStream->raw_Write(var_stream);

Failure : Failed to write binary stream from StorageDriver stream to message stream. AMM Diagnostics Internal error.

Success : Written StorageDriver stream into message stream..

18) spStream->raw_Flush();

Failure : Failed to flush data to the message stream. AMM Diagnostics Internal error.

Success : Flushed data to the message stream.