

Self Service Reset Password Management
COM Object Guide

Version 4.00 (9 February, 2007)

Copyright © Tools4ever 1998 - 2008
<http://www.tools4ever.com>



*Copyright © 1998 - 2007, Tools4ever B.V.
All rights reserved.*

No part of the contents of this user guide may be reproduced or transmitted in any form or by any means without the written permission of Tools4ever.

DISCLAIMER - Tools4ever will not be held responsible for the outcome or consequences resulting from your actions or usage of the informational material contained in this user guide. Responsibility for the use of any and all information contained in this user guide is strictly and solely the responsibility of that of the user.

All trademarks used are properties of their respective owners.

Contents

1.	Introduction	1
<hr/>		
2.	SSRPM COM object reference	2
<hr/>		
2.1.	SSRPM COM object: SSRPM.....	2
2.1.1.	SSRPM.Connect.....	2
2.1.2.	SSRPM.ConnectMultiple.....	2
2.1.3.	SSRPM.GetUserData.....	3
2.1.4.	SSRPM.Enroll.....	3
2.1.5.	SSRPM.GetResetProfileOptions.....	4
2.1.6.	SSRPM.IsBlocked.....	5
2.1.7.	SSRPM.GetResetData.....	5
2.1.8.	SSRPM.CheckAnswers.....	6
2.1.9.	SSRPM.ResetAccount.....	7
2.1.10.	SSRPM.UnlockAccount.....	9
2.1.11.	SSRPM.GetPasswordComplexityRules.....	10
2.2.	SSRPM COM object: SSRPMProfile.....	10
2.2.1.	SSRPMProfile.GetOptions.....	10
2.2.2.	SSRPMProfile.GetNrOfMandatoryQuestionsNeededToEnroll.....	12
2.2.3.	SSRPMProfile.GetNrOfAdminDefinedQuestionsNeededToEnroll.....	13
2.2.4.	SSRPMProfile.GetNrOfUserDefinedQuestionsNeededToEnroll.....	13
2.2.5.	SSRPMProfile.GetMinimumQuestionLength.....	14
2.2.6.	SSRPMProfile.GetMinimumAnswerLength.....	14
2.2.7.	SSRPMProfile.GetMandatoryQuestionsList.....	15
2.2.8.	SSRPMProfile.GetAdminDefinedQuestionsList.....	15
<hr/>		
3.	SSRPM COM with Visual Basic Script	16
<hr/>		
3.1.	Example Visual Basic Script.....	16
3.1.1.	Script section: Connecting to the SSRPM Service.....	17
3.1.2.	Script section: Getting questions from the SSRPM Service.....	17
3.1.3.	Testing and executing the script.....	18
<hr/>		
4.	SSRPM COM with IIS	19
<hr/>		
5.	Appendix A: Component Object Model (COM)	20
<hr/>		
5.1.	COM objects and interfaces.....	20
5.2.	COM registration.....	20
5.3.	Type library.....	20
<hr/>		
6.	Index	21
<hr/>		

1. Introduction

Self Service Reset Password Management (SSRPM) supports Microsoft's Component Object Model (COM, see: *Appendix A: Component Object Model (COM)* on page 20 for more information). This document describes how COM is used in combination with SSRPM. Using COM with SSRPM is referred to as SSRPM COM.

SSRPM COM is a part of Self Service Reset Password Management and installed together with the SSRPM Admin Console. SSRPM COM can be used by client applications to access the SSRPM Service via SSRPM COM objects:

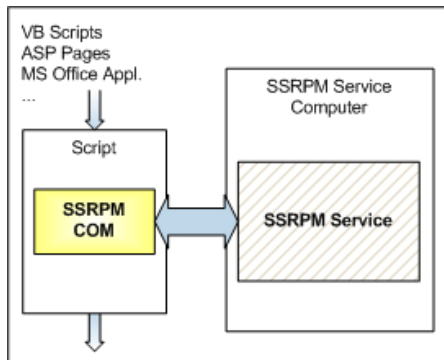


Figure 1: SSRPM Service accessed with SSRPM COM via SSRPM COM objects in a client application

SSRPM COM contains several COM objects, of which the main functions are:

- Enroll a user into SSRPM on the SSRPM Service
- Reset a password of an enrolled user on the SSRPM Service

All SSRPM COM objects are contained in a single file: SSRPMCOM.DLL. This DLL is installed with the SSRPM Admin Console. The COM objects are automatically registered on the computer when the SSRPM Admin Console is installed.

2. SSRPM COM object reference

This chapter describes the all COM objects of SSRPM COM, which are:

- SSRPM: the main COM object;
- SSRPMProfile: stores all SSRPM Profile data and settings.

2.1. SSRPM COM object: SSRPM

This is the main SSRPM COM object. The object is used to connect to an SSRPM Service, access SSRPM user data, reset an account or enroll into SSRPM.

2.1.1. SSRPM.Connect

Purpose: Connect to an SSRPM Service.

Description:

This interface method connects to an SSRPM Service. Usually, this is the first method called after the SSRPM COM object has been created.

Syntax: *Connect([In] String ServerName, [In] Number PortNumber)*

Parameters:

ServerName: The name of the computer that runs the SSRPM Service to which the COM object must connect. The name can be specified as DNS or NETBIOS name.
PortNumber: The port on which the specified SSRPM Service is listening. The default port is 37946 .

Return value: None.

Example (ASP code):

```
Set SSRPM = Server.CreateObject ("SSRPMCOM.SSRPM")  
SSRPM.Connect ("SERVER_A", 37946)
```

2.1.2. SSRPM.ConnectMultiple

Purpose: Connect to the first available SSRPM Service.

Description:

This interface method tries to connect to an SSRPM Service. Usually, this is the first method called after the SSRPM COM object has been created. This method is used if multiple SSRPM Services are configure fail over.

Syntax: *ConnectMultiple([In] StringArray ServerNames, [In] Number PortNumber)*

Parameters:

ServerNames: The name of the computers that run the SSRPM Services to which the COM object must connect. The names can be specified as DNS or NETBIOS names.
PortNumber: The port on which the specified SSRPM Service is listening. The default port is 37946 .

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise.

Example (ASP code):

```
Set SSRPM = Server.CreateObject ("SSRPMCOM.SSRPM")  
Servers[0] = "ServerA"  
Servers[1] = "ServerB"  
SSRPM.ConnectMultiple (Servers, 37946)
```

2.1.3. SSRPM.GetUserData

Purpose: Identify a user and retrieve the applicable SSRPM Profile.

Description:

A connection with the SSRPM Service must be made first with the Connect interface method (see: *SSRPM.Connect* on page 2). The GetUserData interface method retrieves the SSRPM Profile which is applicable for the current user.

Syntax: *GetUserData([In] String Domain, [In] String Account, [In] String Password, [Out] SSRPMProfile Profile)*

Parameters:

Domain: The name of the domain of which the current user is a member of.
Account: The account name of the current user.
Password: The password of the current user.
Profile: A newly created SSRPM Profile object. This object will contain the applicable SSRPM Profile of the current user.

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise.

Example (ASP code):

```
Set Profile = Server.CreateObject("SSRPMCOM.SSRPMProfile")
RetVal = SSRPM.GetUserData("CATS", "John Smith", "#4EdFt6r", Profile)
```

2.1.4. SSRPM.Enroll

Purpose: Enroll a user into SSRPM.

Description:

The GetUserData interface method (see: *SSRPM.GetUserData* on page 3) must be called first to identify a user which must enroll into SSRPM. The Enroll interface method enrolls a user into SSRPM.

Syntax: *Enroll([In] StringArray QuestionList, [In] StringArray AnswerList)*

Parameters:

QuestionList: An array of user questions.
AnswerList: An array of user answers for all specified user questions.

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise.

Example (ASP code):

```
Dim QuestionArray(5), AnswerArray(5)
'Fill arrays with specified questions and answers from user
RetVal = SSRPM.Enroll(QuestionArray, AnswerArray)
```

2.1.5. SSRPM.GetResetProfileOptions

Purpose: Get the enabled SSRPM Profile reset options.

Description:

The GetResetData interface method (see: *SSRPM.GetResetData* on page 5) must be called first to identify the user of which the applicable SSRPM Profile reset options must be retrieved.

The GetResetProfileOptions interface method returns the mask number of all enabled SSRPM Profile reset options. This number identifies which settings are enabled within the current SSRPM Profile. Please refer to 'SSRPMProfile.GetOptions' for a complete list of available objects.

Note: The mask number can be used within a logical And-expression to check if a certain SSRPM Profile option has been enabled. This is shown in the example (ASP Code) below, which checks if the 'Show incorrect answer' option has been enabled.

Syntax: *GetResetProfileOptions()*

Parameters: None.

Return value: The mask number of all enabled SSRPM Profile reset options.

Example (ASP code):

```
ResetProfileOptions = SSRPM.GetResetProfileOptions()  
  
If(ResetProfileOptions And 32) Then  
    response.write("Incorrect answers may be shown.")  
Else  
    response.write("Incorrect answers may not be shown.")  
End if
```

2.1.6. SSRPM.IsBlocked

Purpose: Verify if a user has been blocked by SSRPM.

Description:

Call this method after a call to SSRPM.GetUserData or SSRPM.GetResetData to check if the user is blocked by SSRPM.

Syntax: *IsBlocked()*

Parameters: None.

Return value: 0 if the user is not blocked, 1 if the user is blocked.

Example (ASP code):

```
If(SSRPM.IsBlocked() = 1)  
    response.write("You have been blocked by SSRPM! Please try again later.")
```

2.1.7. SSRPM.GetResetData

Purpose: Identify a user and retrieve all questions the user must answer.

Description:

A connection with the SSRPM Service must be made first with the Connect interface method (see: *SSRPM.Connect* on page 2). The GetResetData interface method retrieves all questions that a user must answer to eventually reset his or her password.

Syntax: *GetResetData([In] String Domain, [In] String Account, [Out] StringArray QuestionList)*

Parameters:

Domain: The name of the domain of which the current user is a member of.
Account: The account name of the current user.
QuestionList: An array which will contain all user questions.

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise.

Example (ASP code):

```
Dim QuestionArray  
RetVal = SSRPM.GetResetData("CATS","John Smith",QuestionArray)
```

2.1.8. SSRPM.CheckAnswers

Purpose: Check answers of a user.

Description:

The GetResetData interface method (see: *SSRPM.GetResetData* on page 5) must be called first to identify the user of which the answers must be checked. The CheckAnswers interface method checks if all questions of a user are answered correctly.

Syntax: *CheckAnswers([In] StringArray QuestionList, [In] StringArray AnswerList, [Out] StringArray InCorrectQuestionList)*

Parameters:

<i>QuestionList:</i>	An array of user questions.
<i>AnswerList:</i>	An array of user answers for all specified user questions.
<i>IncorrectQuestionList:</i>	An array which (optionally) will contain an all questions which are answered incorrectly. This is not the case if the SSRPM Profile option 'Show incorrect answer' has been disabled (see: <i>SSRPM.GetResetProfileOptions</i> on page 4) in the applicable SSRPM Profile.

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise (-11 when the questions are answered incorrectly).

Example (ASP code):

```
Dim QuestionArray(5), AnswerArray(5), InCorrectAnswerArray

'Fill arrays with questions and specified answers from user

RetVal = SSRPM.CheckAnswers(QuestionArray, AnswerArray, InCorrectQuestionArray)

If (RetVal <> 0) Then
    InCorrectQuestionNr = UBound(InCorrectAnswerArray)

    If(InCorrectQuestionNr <> 0) Then
        response.write(→
            "You've answered " & InCorrectQuestionNr & " questions invalid.")
    End if
End If
```

2.1.9. SSRPM.ResetAccount

Purpose: Reset a user password.

Description:

The *GetResetData* interface method (see: *SSRPM.GetResetData* on page 5) must be called first to identify the user of which the password must be reset. The *ResetAccount* interface method resets the password of a user.

Syntax: *ResetAccount*([In] *StringArray* *QuestionList*, [In] *StringArray* *AnswerList*, [In] *String* *NewPassword*, [Out] *StringArray* *InCorrectQuestionList*)

Parameters:

<i>QuestionList:</i>	An array of user questions.
<i>NewPassword:</i>	The new password to which the old password must be set.
<i>AnswerList:</i>	An array of user answers for all specified user questions.
<i>IncorrectQuestionList:</i>	An array which (optionally) will contain an all questions which are answered incorrectly. This is not the case if the SSRPM Profile option 'Show incorrect answer' has been disabled (see: <i>SSRPM.GetResetProfileOptions</i> on page 4) in the applicable SSRPM Profile.

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise (-11 when the questions are answered incorrectly).

Example (ASP code):

```
Dim QuestionArray(5), AnswerArray(5), InCorrectQuestionArray

'Fill arrays with questions and specified answers from user

RetVal = →
SSRPM.ResetAccount(QuestionArray, AnswerArray, "#4EdFt6r", InCorrectQuestionArray)

If (RetVal = 0) Then
    response.write("The password has been reset successfully!")
ElseIf (RetVal = -11) Then
    response.write("The questions are answered incorrectly.")
End If
```

2.1.10. SSRPM.UnlockAccount

Purpose: Unlock a user account.

Description:

The GetResetData interface method (see: *SSRPM.GetResetData* on page 5) must be called first to identify the user of which the account must be unlocked. The UnlockAccount interface method unlocks a user account in the Active Directory.

Syntax: *UnlockAccount*([In] *StringArray* *QuestionList*, [In] *StringArray* *AnswerList*, [Out] *StringArray* *InCorrectQuestionList*)

Parameters:

QuestionList: An array of user questions.
AnswerList: An array of user answers for all specified user questions.
InCorrectQuestionList: An array which (optionally) will contain an all questions which are answered incorrectly. This is not the case if the SSRPM Profile option 'Show incorrect answer' has been disabled (see: *SSRPM.GetResetProfileOptions* on page 4) in the applicable SSRPM Profile.

Return value: Number. 0 if the interface method executed successfully; non-zero otherwise (-11 when the questions are answered incorrectly).

Example (ASP code):

```
Dim QuestionArray(5), AnswerArray(5), InCorrectQuestionArray

'Fill arrays with questions and specified answers from user

RetVal = →
SSRPM.UnlockAccount(QuestionArray, AnswerArray, InCorrectQuestionArray)

If (RetVal = 0) Then
    response.write("The account has been unlocked successfully!")
Else
    response.write("An error has occurred while unlocking the account.")
End If
```

2.1.11. SSRPM.GetPasswordComplexityRules

Purpose: Get the password complexity rules of the domain

Description:

This interface method gets the password complexity rules of the domain from which the enrolled user originates.

Syntax: *GetPasswordComplexityRules*([Out] Number *MinimumPasswordLength*, [Out] Number *PasswordComplexity*)

Parameters:

MinimumPasswordLength The minimum password length.

PasswordComplexity 0 if the password complexity rules are disabled, nonzero if they are enabled.

Return value: 0 if the interface method executed successfully; non-zero otherwise.

Example (ASP code):

```
SSRPM.GetPasswordComplexityRules (MinLength, PasswordComplexity)
```

2.2. SSRPM COM object: SSRPMProfile

This object contains the SSRPM Profile which is applicable for an enrolled user. An SSRPM Profile contains settings like for instance the number of questions a user needs to answer, or the minimum length an answer must be and is applicable for one or more organizational units (OU's) or a whole domain.

See the "Administrator's Guide" for more information about SSRPM Profiles, of which the latest version is available on the *Tools4ever website* <http://www.tools4ever.com>.

2.2.1. SSRPMProfile.GetOptions

Purpose: Get the enabled SSRPM Profile options.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the *GetUserData* interface method (see: *SSRPM.GetUserData* on page 3).

The interface method returns the mask number of all enabled SSRPM Profile options of the SSRPM Profile object. This number identifies which settings are enabled within the current SSRPM Profile. See below for a table with all SSRPM Profile options with their corresponding mask number.

Note: The mask number can be used within a logical And-expression to check if a certain SSRPM Profile option has been enabled. This is shown in the example (ASP Code) below, which checks if Account Blocking has been enabled.

Mask table:

SSRPM Profile Option	Mask Number
A single user cannot have the same answer for multiple questions	1
An answer cannot be a word which is in the corresponding question	2
Case-sensitive comparison of answers	4
Store answers with irreversible encryption	8
Check answers immediately	16
Show incorrect answer	32
Show password complexity rules	64
Hide answers in the SSRPM Enrollment Wizard and SSRPM Reset Wizard	128
Enable Account Blocking	256
Enable E-mail Notification	512
Add an Answer confirmation box to the SSRPM Enrollment Wizard to prevent typo's	1024
Warn users on that they need to re-enroll after a certain period.	2048
Force user enrollment and re-enrollment	4096
Enable minimum answer length	8192

Note: See the "Administrator's Guide" for more information about the SSRPM Profiles options, of which the latest version is available on the *Tools4ever website* <http://www.tools4ever.com>.

Syntax: *GetOptions()*

Parameters: None.

Return value: The mask number of all enabled SSRPM Profile options.

Example (ASP code):

```
ProfileOptions = Profile.GetOptions()  
  
If(ProfileOptions And 256) Then  
    response.write("Account Blocking has been enabled.")  
Else  
    response.write("Account Blocking has not been enabled.")  
End if
```

2.2.2. SSRPMProfile.GetNrOfMandatoryQuestionsNeededToEnroll

Purpose: Get the amount of mandatory administrator defined questions.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the *GetUserData* interface method (see: *SSRPM.GetUserData* on page 3). The interface method returns the number of mandatory administrator defined questions of the SSRPM Profile object.

Mandatory Administrator defined questions are questions which are defined by the administrator *and must be answered by the user*.

Syntax: *GetNrOfMandatoryQuestionsNeededToEnroll()*

Parameters: None.

Return value: The number of mandatory administrator defined questions.

Example (ASP code):

```
AdminQuestions = Profile.GetNrOfMandatoryDefinedQuestionsNeededToEnroll()
```

2.2.3. SSRPMProfile.GetNrOfAdminDefinedQuestionsNeededToEnroll

Purpose: Get the amount of administrator defined questions.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the GetUserData interface method (see: *SSRPM.GetUserData* on page 3). The interface method returns the number of administrator defined questions of the SSRPM Profile object.

Administrator defined questions are questions which are defined by the administrator from which the user must choose when enrolling.

Syntax: *GetNrOfAdminDefinedQuestionsNeededToEnroll()*

Parameters: None.

Return value: The number of administrator defined questions.

Example (ASP code):

```
AdminQuestions = Profile.GetNrOfAdminDefinedQuestionsNeededToEnroll()
```

2.2.4. SSRPMProfile.GetNrOfUserDefinedQuestionsNeededToEnroll

Purpose: Get the amount of user defined questions.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the GetUserData interface method (see: *SSRPM.GetUserData* on page 3). The interface method returns the number of user defined questions of the SSRPM Profile object.

User defined questions are questions which are created by a user itself in which case the user must make up its own questions when enrolling.

Syntax: *GetNrOfUserDefinedQuestionsNeededToEnroll()*

Parameters: None.

Return value: The number of user defined questions.

Example (ASP code):

```
UserQuestions = Profile.GetNrOfUserDefinedQuestionsNeededToEnroll()
```

2.2.5. SSRPMProfile.GetMinimumQuestionLength

Purpose: Get the minimum question length.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the GetUserData interface method (see: *SSRPM.GetUserData* on page 3). The interface method returns the minimum number of characters that a user defined question must contain of the SSRPM Profile object.

Syntax: *GetMinimumQuestionLength()*

Parameters: None.

Return value: The minimum number of characters a user defined question must contain.

Example (ASP code):

```
MinimumQuestionLength = Profile.GetMinimumQuestionLength()
```

2.2.6. SSRPMProfile.GetMinimumAnswerLength

Purpose: Get the minimum answer length.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the GetUserData interface method (see: *SSRPM.GetUserData* on page 3). The interface method returns the minimum number of characters an answer must contain of the SSRPM Profile object.

Syntax: *GetMinimumAnswerLength()*

Parameters: None.

Return value: The minimum number of characters an answer must contain.

Example (ASP code):

```
MinimumAnswerLength = Profile.GetMinimumAnswerLength()
```

2.2.7. SSRPMProfile.GetMandatoryQuestionsList

Purpose: Get a list of all mandatory admin defined questions.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the GetUserData interface method (see: *SSRPM.GetUserData* on page 3). This interface method retrieves a list of all mandatory administrator defined questions of the SSRPM Profile object.

Mandatory Administrator defined questions are questions which are defined by the administrator and must be answered during enrollment.

Syntax: *GetMandatoryQuestionsList([In] String Language, [Out] StringArray QuestionList)*

Parameters:

Language: The language name of the questions. For instance: "English", "French" or "German". If the specified language is not available or empty the default language will be used (English).
QuestionList: An array, which will contain all mandatory administrator defined questions of the SSRPM Profile.

Return value: None.

Example (ASP code):

```
Dim MandatoryQuestionArray
Profile.GetMandatoryQuestionsList "English",MandatoryQuestionArray
```

2.2.8. SSRPMProfile.GetAdminDefinedQuestionsList

Purpose: Get a list of all administrator defined questions.

Description:

An SSRPM Profile object must be created and retrieved from the SSRPM Service first with the GetUserData interface method (see: *SSRPM.GetUserData* on page 3). This interface method retrieves a list of all default administrator defined questions of the SSRPM Profile object.

Administrator defined questions are questions which are defined by the administrator from which the user must choose when enrolling.

Syntax: *GetAdminDefinedQuestionsList([In] String Language, [Out] StringArray QuestionList)*

Parameters:

Language: The language name of the questions. For instance: "English", "French" or "German". If the specified language is not available or empty the default language will be used (English).
QuestionList: An array, which will contain all administrator defined questions of the SSRPM Profile.

Return value: None.

Example (ASP code):

```
Dim AdminQuestionArray
Profile.GetAdminDefinedQuestionsList "English",AdminQuestionArray
```

3. SSRPM COM with Visual Basic Script

A common usage of SSRPM COM is using SSRPM COM in Visual Basic scripts. Visual Basic (VB) is a very general development environment to create Windows applications and scripts. The integration of VB script and SSRPM using SSRPM COM objects allows you to access SSRPM user information, like for instance: the applicable SSRPM Profile settings of a user.

Within this section, SSRPM COM with VB script is described with an example.

3.1. Example Visual Basic Script

This section contains an example of SSRPM COM with VB script. The goal of this example is to show all questions of an enrolled user, which this user has specified within the SSRPM enrollment process.

Note: The script is kept as simple as possible to make it easier to understand and is meant for demonstration purposes.

The example VB script can be found in the '\Examples\VbScript' directory within the SSRPM Admin Console directory (which is by default: 'C:\Program Files\Tools4ever\SSRPM\Admin Console'), called: 'GetQuestions.vbs'.

See below for the complete example script:

```
Option Explicit

Dim SSRPM, QuestionArray, Message, RetVal
Set SSRPM = CreateObject("SSRPMCOM.SSRPM")
RetVal = SSRPM.Connect("LION", 37946)

RetVal = SSRPM.GetResetData("CATS", "John Smith", QuestionArray)

If RetVal = 0 Then

    For Each Question In QuestionArray
        Message = Message & Question & Chr(13)
    Next

    MsgBox Message, VbOkOnly, "SSRPM COM with VB Script Example"

Else

    MsgBox "An error occurred while getting questions from user (code: " & RetVal & ").", VbOkOnly, "Error getting questions"

End If
```

In the next sections, parts of the scripts are shown with some comments for each section.

3.1.1. Script section: Connecting to the SSRPM Service

Before the question can be get and shown a connection with the SSRPM Service needs to be setup first. This is done by creating the 'SSRPM' COM object of SSRPM COM and calling the 'Connect' interface method of the object.

First, the needed variables are declared:

```
Dim SSRPM, QuestionArray, Message, RetVal
```

The 'SSRPM' variable will hold the SSRPM COM object. The variable 'QuestionArray' will hold an array of text values with all user questions. The 'Message' variable will hold the text value of all questions together and will be used to show all questions in a message box and the 'RetVal' variable will be used to store the return value which can be returned by an interface method.

The SSRPM COM object SSRPM is then created with the 'CreateObject' call:

```
Set SSRPM = CreateObject("SSRPMCOM.SSRPM")
```

The argument 'SSRPMCOM.SSRPM' specifies the SSRPM COM library and the type of object (SSRPM) of which an instance must be created. If the SSRPM COM library is not installed and/or registered, the 'CreateObject' call will fail.

The SSRPM COM object now connects to the SSRPM Service with the statement:

```
RetVal = SSRPM.Connect("LION", 37946)
```

The 'Connect' interface method takes two arguments: the name of the computer that runs the SSRPM Service and the port number used by the SSRPM Service. 37946 is the default SSRPM Service port number.

3.1.2. Script section: Getting questions from the SSRPM Service

The SSRPM COM object is connected to the SSRPM Service. Now the questions can be retrieved from the SSRPM Service by calling the 'GetResetData' interface method of the SSRPM COM object. The SSRPM COM object gets all questions from the specified user:

```
RetVal = SSRPM.GetResetData("CATS", "John Smith", QuestionArray)
```

The 'GetResetData' interface method takes three arguments: the name of the domain of which the enrolled user is a member of, the account name of the enrolled user and the declared 'QuestionArray' variable. On success, the return value of the 'GetResetData' interface method is zero and the 'QuestionArray' variable will contains a text array of all user questions.

A loop is implemented to copy each question from the returned question array within the 'QuestionArray' variable to one text variable ('Message'):

```
For Each Question in QuestionArray  
    Message = Message & Question & Chr(13)  
Next
```

Each question is separated with a new line, by using carriage return characters ('Chr(13)') between each question within the 'Message' variable. The 'Message' variable is used to show all questions in a message box:

```
MsgBox Message, VbOkOnly, "SSRPM COM with VB Script Example"
```

3.1.3. Testing and executing the script

To test the script in your environment, modify the Visual Basic script with your favorite editor. Make sure that the user and domain name is a user that has been enrolled into SSRPM and that the SSRPM COM object connects to the appropriate SSRPM Service.

To eventually execute the script, open a command prompt and enter the name of the script: *GetQuestions.vbs*.

This will automatically initialize Windows Scripting Host to execute the script. When the script executes successfully a message box will be prompted, which shows all questions of the specified user:

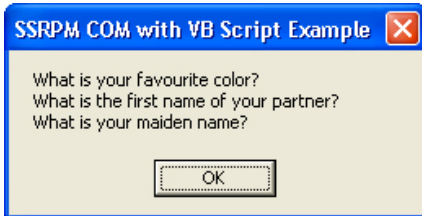


Figure 2: VB Script example output

4. SSRPM COM with IIS

Users can use SSRPM with an internet browser via SSRPM COM objects in Advanced Server Pages (ASP) with Internet Information Services (IIS).

In such an environment three main components are involved:

1. The internet browser that shows the web-pages and is used to enter input fields;
2. The IIS web-server, which hosts the ASP pages and to which the browser connects;
3. The SSRPM Service, contacted by the SSRPM COM objects used by the ASP pages running on the IIS web-server.

See the figure below how these components interact with each other:

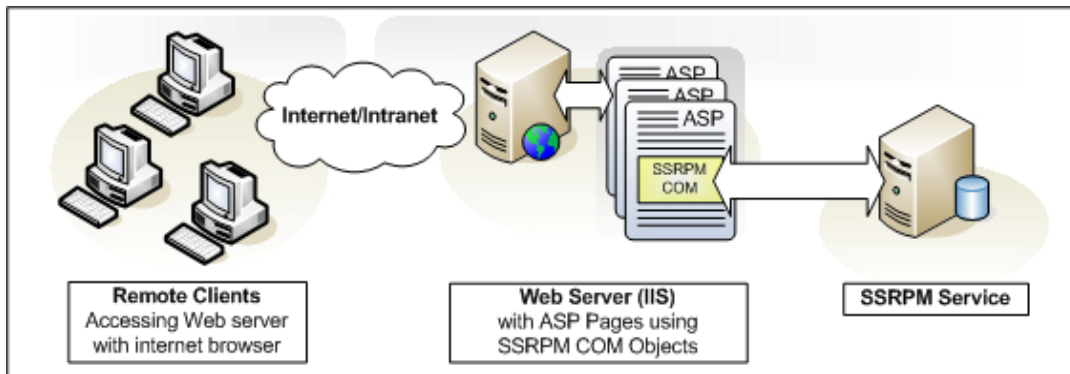


Figure 3: SSRPM COM with IIS and ASP Pages

SSRPM is shipped with a fully functional web interface. Please refer to the document 'Web Interface Guide' for a complete list of features and how to install and configure the SSRPM Web Interface.

5. Appendix A: Component Object Model (COM)

The Component Object Model (COM) is a technology that is used by applications to interact with other applications. The COM technology is designed by Microsoft. The most important Microsoft applications that use COM are:

- Internet Information Services (IIS)
- Office applications
- Visual Basic Scripting (VB) and Visual Basic

5.1. COM objects and interfaces

In principle, a COM object is a piece of software that implements one or more functions. Different COM objects support different functions. The functions of a COM object are accessible by means of the interface of the COM object. The COM object itself is regarded as a black box that implements one or more functions accessible through its interfaces.

Applications that support COM can create COM objects. By accessing the interface functions of the COM object, the functions of the COM object are executed by the calling application. The syntax to create COM objects and access the interface functions of a COM object is extremely general: this is the main reason why COM objects can be used by so many different applications: The same COM object can be created and used in ASP-pages, Word documents and Visual Basic scripts.

All applications that support COM use some kind of programming or script language to implement COM. The procedure used is always the same:

1. The COM object is created;
2. The interface functions of the COM object are accessed;
3. Returned variables can be processed in the application.

An application can use multiple COM objects and COM objects can use other COM objects.

5.2. COM registration

In order for an application to use a COM object and the interface functions of the object, the COM object must be registered. Once a COM object is registered, all applications that support COM can use the COM object. In most cases, COM objects are registered automatically.

5.3. Type library

In most cases, the COM object code is contained in a file with the .DLL extension. Such a file contains all code needed to use the COM objects it implements. Besides the COM objects, the file can also contain a so called type library that describes the COM objects and the interfaces that are used to access the COM objects.

6. Index

A

Appendix A
Component Object Model (COM) • 1, 20

C

COM objects and interfaces • 20
COM registration • 20

E

Example Visual Basic Script • 16

I

Introduction • 1

S

Script section
Connecting to the SSRPM Service • 17
Getting questions from the SSRPM Service • 17
SSRPM COM object
SSRPM • 2
SSRPMProfile • 10
SSRPM COM object reference • 2
SSRPM COM with IIS • 19
SSRPM COM with Visual Basic Script • 16
SSRPM.CheckAnswers • 6
SSRPM.Connect • 2, 3, 5
SSRPM.ConnectMultiple • 2
SSRPM.Enroll • 3
SSRPM.GetPasswordComplexityRules • 10
SSRPM.GetResetData • 4, 5, 6, 7, 9
SSRPM.GetResetProfileOptions • 4, 7, 9
SSRPM.GetUserData • 3, 10, 12, 13, 14, 15
SSRPM.IsBlocked • 5
SSRPM.ResetAccount • 7
SSRPM.UnlockAccount • 9
SSRPMProfile.GetAdminDefinedQuestionsList • 15
SSRPMProfile.GetMandatoryQuestionsList • 15
SSRPMProfile.GetMinimumAnswerLength • 14
SSRPMProfile.GetMinimumQuestionLength • 14
SSRPMProfile.GetNrOfAdminDefinedQuestionsNeeded
ToEnroll • 13
SSRPMProfile.GetNrOfMandatoryQuestionsNeededToE
nroll • 12
SSRPMProfile.GetNrOfUserDefinedQuestionsNeededTo
Enroll • 13
SSRPMProfile.GetOptions • 10

T

Testing and executing the script • 18
Type library • 20