

DotNet Connector Development Manual

Table of Contents

ACRONYMS AND ABBREVIATIONS.....	8
PART I: INTRODUCTION	9
CHAPTER 1 INTRODUCTION	10
1.1 ROLES AND RESPONSIBILITIES OF STAKEHOLDERS	10
1.2 PRE-REQUISITES	10
1.3 HOW TO USE THIS MANUAL.....	11
1.4 MOTIVATION FOR CONNECTORS	12
CHAPTER 2 ALL ABOUT CONNECTORS.....	13
2.1 REGISTRATION	13
2.1.1 SAP Registration.....	14
2.1.2 SAP Register Service	15
2.1.3 SP Registration.....	16
2.1.4 SP Register Service.....	17
2.2 TYPES OF CONNECTORS	18
2.2.1 WHY TWO TYPES OF CONNECTORS?.....	19
2.2.2 Generic Connector	19
2.2.2.1 RESPONSIBILITIES OF SAP GENERIC CONNECTOR.....	19
2.2.2.2 RESPONSIBILITIES OF SP GENERIC CONNECTOR.....	20
2.2.3 Application Specific Connector	20
2.2.3.1 RESPONSIBILITIES OF SAP APPLICATION SPECIFIC CONNECTOR.....	20
2.2.3.2 RESPONSIBILITIES OF SP APPLICATION SPECIFIC CONNECTOR.....	21
2.2.4 Synchronous and Asynchronous Request.....	21
2.3 WORKING WITH CONNECTORS	21
2.3.1 Connectors in Simple Gateway Structure.....	21
2.3.2 Connectors in a Constellation of Gateways.....	24
PART II: TECHNICAL DETAILS.....	27
CHAPTER 3 GENERIC CONNECTOR.....	28
3.1 AUTHENTICATION IN CONNECTORS.....	28
3.2 SAP GENERIC CONNECTOR.....	29
3.2.1 Responses that SAP receives	29
3.2.2 SAP Generic Connector Functions.....	29
3.2.3 SAP Generic Connector Properties.....	30
3.2.4 SAP Generic Connector Functions.....	31
3.2.4.1 SEQUENCE FOR ASYNCHRONOUS SUBMIT REQUEST	31
a) Asynchronous Submit Request.....	34
b) Document Poll.....	36
c) Document Delete	37
d) Document List.....	38
3.2.4.2 SEQUENCE FOR SYNCHRONOUS SUBMIT REQUEST	40
a) Synchronous Submit Request.....	41
3.3 SP GENERIC CONNECTOR	44
3.3.2 SP Generic Connector Properties.....	44
3.3.3 Details of SP Generic Connector Functions.....	45
CHAPTER 4 APPLICATION CONNECTOR.....	47
4.1 SAP APPLICATION CONNECTOR.....	47

.Net Connector Development Manual

4.2	SP APPLICATION CONNECTOR	47
	a) <i>Asynchronous Submit Request</i>	48
	b) <i>Synchronous Submit Request</i>	48
PART III:	CONNECTOR TROUBLESHOOTING	50
CHAPTER 5	TESTING AND TROUBLESHOOTING.....	51
5.1	SAP CONNECTOR.....	51
5.2	SP CONNECTOR.....	62
5.3	ERROR LOGS	68
APPENDIX A:	ERROR CODES	68
APPENDIX B:	APP.CONFIG FILE FOR SAP GENERIC CONNECTOR.....	75
APPENDIX C:	WEB.CONFIG FILE FOR SP GENERIC CONNECTOR.....	76
APPENDIX D:	ERROR SCHEMA.....	80
	BUSINESS ERROR RESPONSE SCHEMA	80
	LIST RESPONSE SCHEMA	81
	LIST REQUEST SCHEMA.....	82

List of Tables

Table 1 : SAP Generic Connector Properties	30
Table 2 : Input Parameters for Asynchronous Submit Request	34
Table 3 : Output Parameters for AsynchronousSubmitRequest in case of Response	35
Table 4 : Input Parameters for Asynchronous Submit Request with SHAI encoding	35
Table 5 : Output Parameters for AsynchronousSubmitRequest in case of Response	35
Table 6 : Input Parameters for Asynchronous Submit Request with digital signature	36
Table 7 : Output Parameters for Asynchronous Submit Request in case of Response	36
Table 8 : Document Poll Input Parameters	36
Table 9 : Document Poll Output Parameters.....	37
Table 10 : Document Delete Input Parameters	37
Table 11 : Document Delete Output Parameters	37
Table 12 : Document List Input Parameters	38
Table 13 : Document List Output Parameters	38
Table 14 : Document List Input Parameters with SHAI encoding	38
Table 15 : Document List Output Parameters	39
Table 16 : Document List Input Parameters with Digital Signature	39
Table 17 : Document List Output Parameters	40
Table 18 : Input Parameters for Synchronous Submit Request	41
Table 19 : Output Parameters for Synchronous Submit Request in case of Response	42
Table 20 : Input Parameters for Synchronous Submit Request with SHAI encoding	42
Table 21 : Output Parameters for Synchronous Submit Request in case of Response	42
Table 22 : Input Parameters for Synchronous Submit Request with Digital Signature	43
Table 23 : Output Parameters for Synchronous Submit Request in case of Response	43
Table 24 : SP Generic Connector Properties	44
Table 25 : Submit Response Input Parameters	45
Table 26 : Submit Response Output Parameters	46
Table 27: Asynchronous Submit Request Input Parameters	48
Table 28 : Synchronous Submit Request Input Parameters	48
Table 29 : Synchronous Submit Request Output Parameters	49

Table of Figures

<i>Figure 1 : Block Diagram of Connector Architecture</i>	12
<i>Figure 2 : SAP Registration</i>	15
<i>Figure 3 : SAP Register Service</i>	16
<i>Figure 4 : SP Registration</i>	17
<i>Figure 5 : SP Service Registration</i>	18
<i>Figure 6 : Connectors in Simple Gateway Structure</i>	22
<i>Figure 7 : Connectors in a Constellation of Gateways</i>	24
<i>Figure 8: Sequence Diagram for Asynchronous Request</i>	32
<i>Figure 9 : Sequence Diagram for Synchronous Request</i>	40

Acronyms and Abbreviations

Service Access Providers (SAP): A SAP provides easy access to services provided by the government to service seekers. Examples of SAP are front-end infrastructure (web-portal), a kiosk in rural area or a Citizen Service Centre (CSC).

Citizen Service Centre (CSC) : Service access point for the citizens.

Service Provider (SP): SP is a back-end government department or any other third-party agency offering e-services to citizens, businesses and to other government departments.

State e-governance Service Delivery Gateway (SSDG): SSDG is a service exchange infrastructure that routes the services desired by the end user to the appropriate service provider (state department). It thus acts as a hub for e-governance services to be functional at state level.

Interoperability Interface Protocol/Specification (IIP/S): The protocol used to communicate with the Gateway by the client application such as SAP and SP.

Implementation Agency (IA): Agencies which are appointed by the States to deploy SSDG and develop application specific connector to connect the applications to SSDG. The implementation agencies are to be chosen from the list of the DIT empanelled agencies for eforms on State Portal and SSDG project.

Part I: Introduction

Chapter 1 Introduction

This manual is intended for the developers who want to build an application specific connector to communicate with SSDG using the .Net framework.

1.1 Roles and Responsibilities of Stakeholders

The stakeholders in this project are C-DAC, empanelled implementation agencies and the state governments.

1.1.1 State government

- The state government is the one who initiates the deployment of SSDG and owns it.
- It acquires the gateway from C-DAC through DIT.

1.1.2 Implementation Agency

- Will be deploying the SSDG.
- Will be developing the connectors for applications given by state government departments to communicate with SSDG.
- Will be providing the maintenance for the same for three years.

1.1.3 C-DAC

- Will provide the SSDG software product and the SSDG stack
- Will provide the Bill of Material required for SSDG
- Will provide the patches for the SSDG software product
- Will provide the centralized support for the SSDG software product
- Will provide the manuals
- Consultancy for developing the application specific connectors to the implementation agencies will be provided by C-DAC. ????
- Generic connectors will be provided by C-DAC.

1.2 Pre-requisites

User proficiency in



- i. Proficiency in .Net Framework 2.0 and above.
- ii. XML Serialization and De-serialization.
- iii. Communicating with Web services developed in .Net Framework.

1.3 How to use this Manual.

This manual is divided into three sections each explaining different aspect of the connector development.

Note: Users who are well acquainted with the basic concepts and working of connectors, may skip Part 1 and directly refer Part 2 for technical details.

Part 1 (Introduction)

Part One provides an introduction to connectors. It gives an overview about connectors and their role in communication between the SAP and the SP. It also lists the types of connectors and their positioning in the SSDG architecture.

Chapter 1: Covers the prerequisites for connector development. Provides details about the types of connectors and their significance in SSDG. Also covers roles and responsibilities of various stakeholders.

Chapter 2: Extensively describes the significance, functionality and working of the connectors in SSDG architecture.

Part 2 (Technical Details)

It covers the technical aspect of the connectors. Explanation of various features and functionality of the connectors is provided in this section.

Chapter 3: Describes the authentication mechanism for connectors as well as the details of generic connectors.

Chapter 4: Covers development of application specific connectors.

Part 3(Troubleshooting)

This part deals with troubleshooting related to connector development. It lists out the various error codes and logs.

Chapter 5: Lists out the various error codes that may be faced during the development of application specific connectors. It also explains the significance of maintaining error log for efficient trouble shooting and problem fixing.

1.4 Motivation for Connectors

The core functionality of the SSDG is to provide messaging communication between the SAP and the SP and vice versa for a citizen to avail the various services offered by SP. The citizen makes use of SAP to avail services offered by SP. The SAP and SP in turn communicate with each other through SSDG for rendering services to the citizen. In order to accomplish this task, citizen's request should be in the format understandable to SSDG. Here the connectors come into picture. Connectors are nothing but a piece of software that channels the data between SAP and SP through SSDG by manipulating the data as per standards and specifications of SSDG and later of SP. Connectors may also be looked upon as an integration mechanism between SAP and SP and SSDG

Connectors, as the name suggests, serve the basic purpose of connecting Service Access Provider (SAP) and Service Provider (SP) to State e-governance Service Delivery Gateway (SSDG).

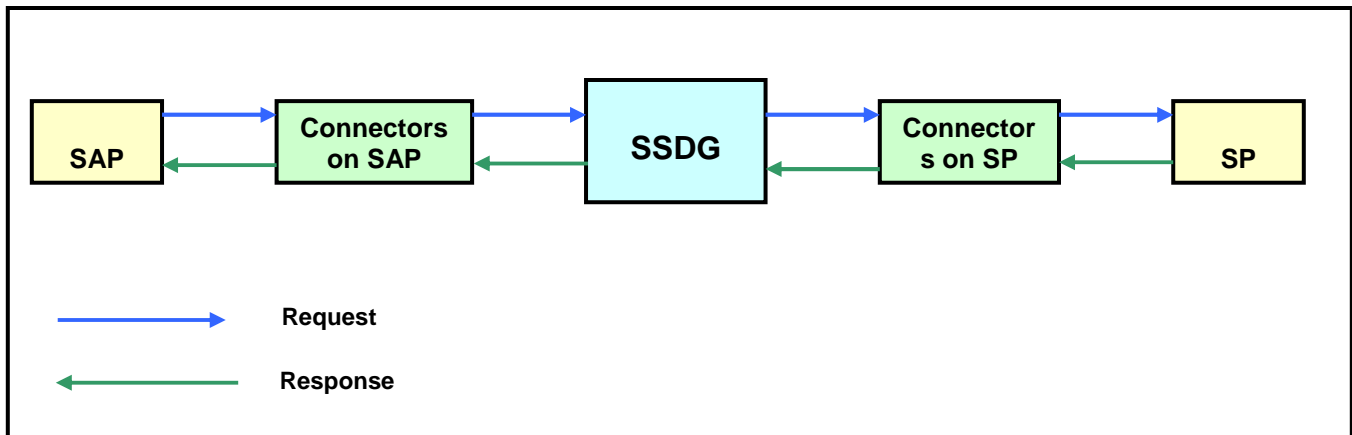


Figure 1 : Block Diagram of Connector Architecture

A connector is essentially a software (a piece of code in the form of DLL or Web-Service) acting as an adapter to communicate with SSDG. The messages that are exchanged between the SAP and SP applications through SSDG pass through the connector first i.e. SAP and SP send and receive messages through SSDG by using the services of the connector.

On SAP as well as SP side, there is a requirement for two units of connectors to be operational viz., *Generic connector* and *Application Specific Connector*. The application specific connector uses the APIs provided by the generic connector for passing the data provided by SAP and SP on either side. Whereas, the generic connector which is provided by C-DAC for both SAP and SP side, interfaces

with the gateway and forwards/receives this data to/from the gateway by complying to Gateway Interoperability standards (IIP/IIS).

Chapter 2 All About Connectors

This chapter covers the details about the communication between the SAP and SP that occurs for accessing and providing a service through SSDG. The role of connectors in this exchange is covered in this chapter.

2.1 Registration

Before getting into the details of the connector, one needs to be familiar with the functioning of SAP and SP. It is important to note that SAP needs to be registered with SSDG before availing the services of the SP. Similarly, SP needs to register with SSDG to provide its services to SAP.

Also, it is equally mandatory for the SP to register each of its services individually with SSDG. Similarly, the SAP has to register for each service individually for accessing those services.

Note: SAP needs to be registered to SSDG to avail any services. SP and the services offered should be separately registered to SSDG for providing these services on SSDG

2.1.1 SAP Registration

The SAP has to register with the SSDG for availing services of any SP. For this, the SAP needs to fill up the **SAP Registration** form on the relevant State Portal giving details like Organisation Name, Description, Valid Upto, Contact Person, Address, City, State, Pin Code, Phone No and E-mail address. Once the SAP is registered, an SAP ID along with a Password shall be sent through e-mail to SAP. SAP can now avail of various services using this SAP ID and Password.

Welcome
KAPIL KANT KAMAL

Reset Password

- Reset Password

PMU Project Board

- List of Gateway Services
- SAP Registration Report
- SP Enrollment Report
- SAP Deregistration Report
- SP Deenrollment Report
- Gateway Utilization Report
- Gateway Throughput Report
- Service Utilization Report

SAP

- SAP Profile Update
- SAP Deletion
- SAP Register Service
- De-Register SAP Registered Service
- SAP Registration

SAP Registration Form

Organization Name *	<input type="text"/>
Description*	<input type="text"/>
Valid Upto	<input type="text"/>
Contact Person *	<input type="text"/>
Address *	<input type="text"/>
City *	<input type="text"/>
State *	--STATE--
Pin code *	<input type="text"/>
Phone No *	<input type="text"/>
	(WITH STD CODE)
Email Address *	<input type="text"/>

Submit Reset Home

Click on the Reset button to clear the fields.

Terms of Use |Disclaimer |Designed managed and maintained by www.cdacmumbai.in © 9IT

Figure 2 : SAP Registration

2.1.2 SAP Register Service

As mentioned earlier, along with registering itself with SSDG, SAP also needs to register the services it wants to avail. For this, SAP has to fill up **SAP Register Service Form** on the relevant State Portal enlisting details like Service ID (ID of the service SAP desires to avail), SAP ID and Validation date (date upto which the services can be accessed by SAP) for accessing that service.

The screenshot displays the NSDG (National e-Governance Service Delivery Gateway) interface. At the top, the header includes the NSDG logo and the text 'National e-Governance Service Delivery Gateway' and 'A Messaging Middleware for Integration & Interoperability'. Below the header, there are navigation links: HOME, CONTACT US, SITE MAP, and LOGOUT. The main content area is divided into several sections:

- Welcome KAPIL KANT KAMAL**
- Reset Password** section with a 'Reset Password' link.
- PMU Project Board** section with links for: List of Gateway Services, SAP Registration Report, SP Enrollment Report, SAP Deregistration Report, SP Deenrollment Report, Gateway Utilization Report, Gateway Throughput Report, and Service Utilization Report.
- SAP** section with links for: SAP Profile Update, SAP Deletion, **SAP Register Service** (highlighted), De-Register SAP Registered Service, and SAP Registration.

The **SAP Register Service Form** is the central focus, containing the following fields:

Service ID *	<input type="text" value="http://202.141.151.140/31"/>
Sap ID *	<input type="text" value="SAP080920081104"/>
Valid Upto	<input type="text" value="10/31/2008"/>

Below the form are three buttons: Submit, Reset, and Home. A callout box points to the 'Reset' button with the text: 'Click on the Reset button to clear the fields.'

At the bottom of the page, there is a footer with the text: 'Terms of Use | Disclaimer | Designed managed and maintained by www.cdacmumbai.in © 2007'

Figure 3 : SAP Register Service

2.1.3 SP Registration

Similarly, SP also has to register with SSDG for providing its services. For this, SP needs to fill **SP Enrolment Form** on the relevant State Portal providing following details:

Organisation Name, Description, Contact Person, Address, City, State, Pin Code, Phone No and E-mail address. Once SP has registered, an SP ID along with a Password shall be sent through e-mail to SP.

Organization Name *

Description *

Contact Person *

Address *

City *

State * —STATE—

Pin code *

Phone No *

Email Address *

Submit Reset Home

Terms of Use | Disclaimer | Designed managed and maintained by www.cdacnubel.in © DIT

Figure 4 : SP Registration

On completion of all the above mentioned steps, SP will be designated as registered service provider.

2.1.4 SP Register Service

SP needs to register its services with SSDG for making them available to SAP. Each and every service of SP has to be enrolled separately by providing details of the service. To do so, SP needs to fill up *Service Enrolment Form* on the relevant State Portal. SP has to give the details such as Service Name, Service URL, Description, Method URL, Method Name, Domain, Authentication Type, Response Type, Poll Interval, SP ID. Upon registering with SSDG, a unique Class ID is assigned to the service registered by SP. This Class ID is critical in resolving services when a request for a particular service is raised.

Note: *Authentication Type* determines how SAP will be authenticated by SP. Note that when SAP makes a request to SP, the SAP application connector should ensure that the property

pertaining to the Authentication Mode is set as per the requirement of service registered by SP.

Response Type determines how SP will respond to the request by SAP. While making request, the SAP application connector sets this property as per the requirement of service registered by SP. There are 2 types of response based on the 2 types of requests. In case of synchronous request, the response is served in a single cycle whereas in case of asynchronous request the response submitted using the API of SP generic connector once SP application process the request.

The screenshot shows the NSDG Service Enrolment Form. The header includes the CDCC logo and the text 'National e-Governance Service Delivery Gateway' and 'NSDG'. A navigation bar contains 'HOME', 'CONTACT US', 'SITE MAP', and 'LOGOUT'. The left sidebar has a 'Welcome KAPIL KANT KAMAL' message and a menu with options like 'Reset Password', 'SP', 'Gateway Utilization Report', 'Search For Services', 'List Services', 'Service Registration', 'SP Deletion', 'SP Updation', 'SP Registration', 'Service Updation', and 'Service De-Registration'. The main form area is titled 'Service Enrolment Form' and contains the following fields:

Service Name *	<input type="text"/>
Service URL *	<input type="text"/>
Description *	<input type="text"/>
Method URL *	<input type="text"/>
Method Name *	<input type="text"/>
Domain *	<input type="text"/>
Authentication Type *	--Authentication Type--
Response Type *	<input type="radio"/> Synchronous <input checked="" type="radio"/> Asynchronous
Poll Interval	<input type="text"/>
SP ID *	<input type="text"/>

Below the form are 'Submit', 'Reset', and 'Home' buttons. A yellow callout box points to the 'Reset' button with the text 'Click on the Reset button to clear the fields'. The footer contains 'Terms of Use | Disclaimer | Designed managed and maintained by www.cdacmumbai.in © DIT'.

Figure 5 : SP Service Registration

2.2 Types of Connectors

There are two types of connectors operational on the SP and SAP side,

- i. Generic Connector.
- ii. Application Specific Connector.

2.2.1 Why Two Types of Connectors?

Connectors need to cater two types of functionalities (i) common tasks that are required for

communicating with the SSDG and (ii) functionalities to communicate with SP. While generic connector provides functionalities for some common tasks associated with Gateway communication, the application specific connector is responsible for the functionalities related to application needs.

It also provides with separation of concern between Implementation Agency and C-DAC. C-DAC will own up responsibilities towards Generic connector whereas provisioning and maintenance of Application Specific connector will be Implementation Agency's responsibility.

There are some common or general requirements that are to be fulfilled for communicating with Gateway to comply with the gateway specifications which are taken care in the generic connectors. The application level requirements vary depending on factors like application vendors, the application development platform, hence there is a need for application specific connector to fulfil these requirements at application level leading to two separate connectors, generic and application specific, both at SAP and SP side.

2.2.2 Generic Connector

Generic connector consists of a set of APIs (*ConnectorInterface class in case of SAP*) that are exposed to the application specific connector. Application specific connectors should use these APIs to communicate with generic connectors. Generic connector packages the request in the required format of SSDG.

2.2.2.1 Responsibilities of SAP Generic Connector

- i.) Request packaging: When a request is sent from SAP to SP for availing the services of SP, SAP generic connector converts this request message for compliance with the Interoperability protocol (IIP) used by Gateway.
- ii.) Web service communication: in IIS compliant format.
- iii.) 3 modes of Authentication support: SAP generic connector also provides authentication features for validating SAP to SP through several authentication mechanisms like *SHA1 encoding, Digital Certificates or Clear Text*. The type of authentication depends upon the services availed by SAP and decided a priori by SAP and SP.
- iv.) Provision for data signing for integrity check: SAP generic connector will aid the integrity check of the request message by signing the request message with digital

signature. This shall be validated at the SP generic connector end to ensure that the request message has not been modified or tampered with during transit.

2.2.2.2 Responsibilities of SP Generic Connector

- i) Web service interface: SP Generic connector is a webservice interface (compliant to IIS) and receives the request message from SSDG in synchronous /asynchronous mode and responds accordingly.
- ii) Request extraction: It extracts the payload, the *CorrelationId*, *Transaction Id* and forwards the same to SP application by invoking the appropriate function of the SP application specific connector.
- iii) Provision for integrity check: SP generic connector has a provision to carry out end-to-end integrity checks to ensure that the request message has not been modified or tampered with during transit.
- iv) Web service client: It also provides the API for SP application connector to submit the response generated by SP application. The SP application connector uses this API to submit response to SP generic connector which is then forwarded to SSDG.

2.2.3 Application Specific Connector

An application specific connector should use the APIs exposed by the generic connector to communicate with SSDG. Application specific connector should convert the message that is understandable to SP.

2.2.3.1 Responsibilities of SAP Application Specific Connector

- i) Object to XML serialization: When a request is sent from SAP to SP for availing the services of SP, SAP application connector converts this request message in XML format for compliance with format specified by SP for availing its services.
- ii) Service setup: It is also the responsibility of the SAP application connector to search for the *Class ID* (*Class Id* is the Id of the service that SAP wishes to avail. This is assigned to the service by the Gateway when the service is registered with the gateway) pertaining to a specific service for availing this service.
- iii) Generic Connector API invocation: The SAP application connector invokes appropriate functions of the SAP generic connector API for availing services of SP.

2.2.3.2 Responsibilities of SP Application Specific Connector

- i) Preparation of call handler: It implements the interfaces defined by SP generic connector.
- ii) Mapping of ClassID to method implementation: The SP application connector is also responsible for looking up the actual methods that SP uses to process the request and generate the response. These methods are identified based on their class ids.
- iii) Generic connector API invocation: Once identified, the SP application specific connector calls these methods for processing the request and sends back the response to SP generic connector to complete the cycle for synchronous / asynchronous message.

2.2.4 Synchronous and Asynchronous Request

Before we proceed with the details about how connectors work, what their significance is, when their services are used etc, we should be familiar with the types of requests that are involved in the communication between SAP and SP. There are two types of requests possible (i) Synchronous and (ii) Asynchronous.

Synchronous Request – If a request made by SAP is served by SP without breaking the connection, then it is a Synchronous Request.

Asynchronous Request – If a request made by SAP is served by SP at some later point of time, then such request is termed as Asynchronous Request. The connection breaks down after the request is made and response to the request is sent at a later stage. Request and response are separate execution cycles in the case of asynchronous communication.

2.3 Working with Connectors

The SSDG architecture is explained below. Here we see how SAP and SP communicate with each other through SSDG and what role the connectors have to play in facilitating this communication.

2.3.1 Connectors in Simple Gateway Structure

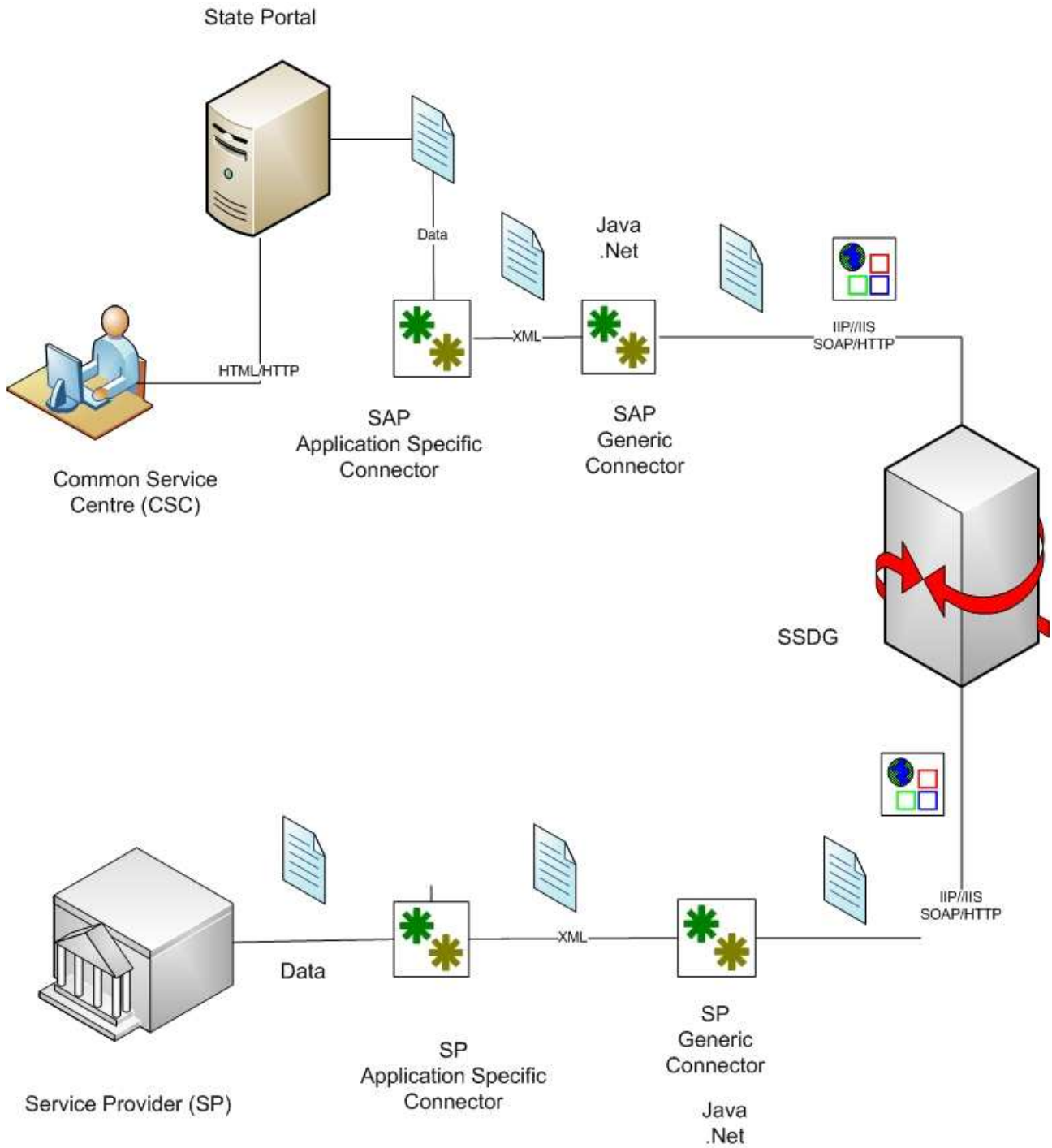


Figure 6 : Connectors in Simple Gateway Structure

Conceptual Working of Connectors on SAP side

Request from SAP to avail SP service is sent through SSDG. Following steps explain the actions taking place at SAP side in a sequential order.

- a. SAP application sends the request to SAP application specific connector.
- b. The message request sent by SAP application specific connector should be in the XML format as per the XSD (schema) defined by SP.
- c. Application specific connector uses the API i.e. class *ConnectorInterface* of the SAP generic connector to pass the request in XML format to SP side through SSDG.
- d. The generic connector converts the payload as per the IIP/IIS standards before sending it to the SSDG.
- e. If the request is synchronous, the following steps are executed to serve the request.
 - Suppose a SAP requests for the PAN details of a citizen by entering his/her PAN card number.
 - This request will be served by SSDG instantaneously by fetching the PAN details of the citizen from SP based on his/her Pan card number.
 - Thus details are available to SAP in real time basis in a single continuous cycle. This is how a synchronous request is executed on SSDG.
- f. If the request is asynchronous, SSDG returns an acknowledgement for the same to SAP. The acknowledgement contains an ID (*correlationId* generated by SSDG) for the request. Using this ID, the citizen can track the status of his request by using poll operation on SSDG. The following steps are executed to serve the request but response from SP is not received immediately unlike in synchronous request

Conceptual Working of Connectors on SP side

The following business process is catered on SP side

- a. The SP generic connector checks whether the request message sent by SSDG is valid and takes appropriate action in case of valid/invalid request messages.
- b. If the message sent from SSDG is invalid then the SP generic connector sends the message back to SSDG along with the error message conveying appropriate reason for the error
- c. .If the message sent by SSDG is valid, the SP generic connector passes the message to the SP application specific connector by extracting the payload and other necessary details.
- d. Based on the payload and details provided by SP generic connector, SP application specific connector sends the request to SP application whose service is requested.
- e. SP application specific connector then formats and sends back the response returned by SP application to SP generic connector using the SP generic connector API.(for asynchronous request, the response is returned using generic connector API whereas in case of synchronous request the response is sent in a single call to SP)

- f. The generic connector converts the payload as per the IIP/IIS standards before sending it to the SSDG which in turn sends it to SAP generic connector.

2.3.2 Connectors in a Constellation of Gateways

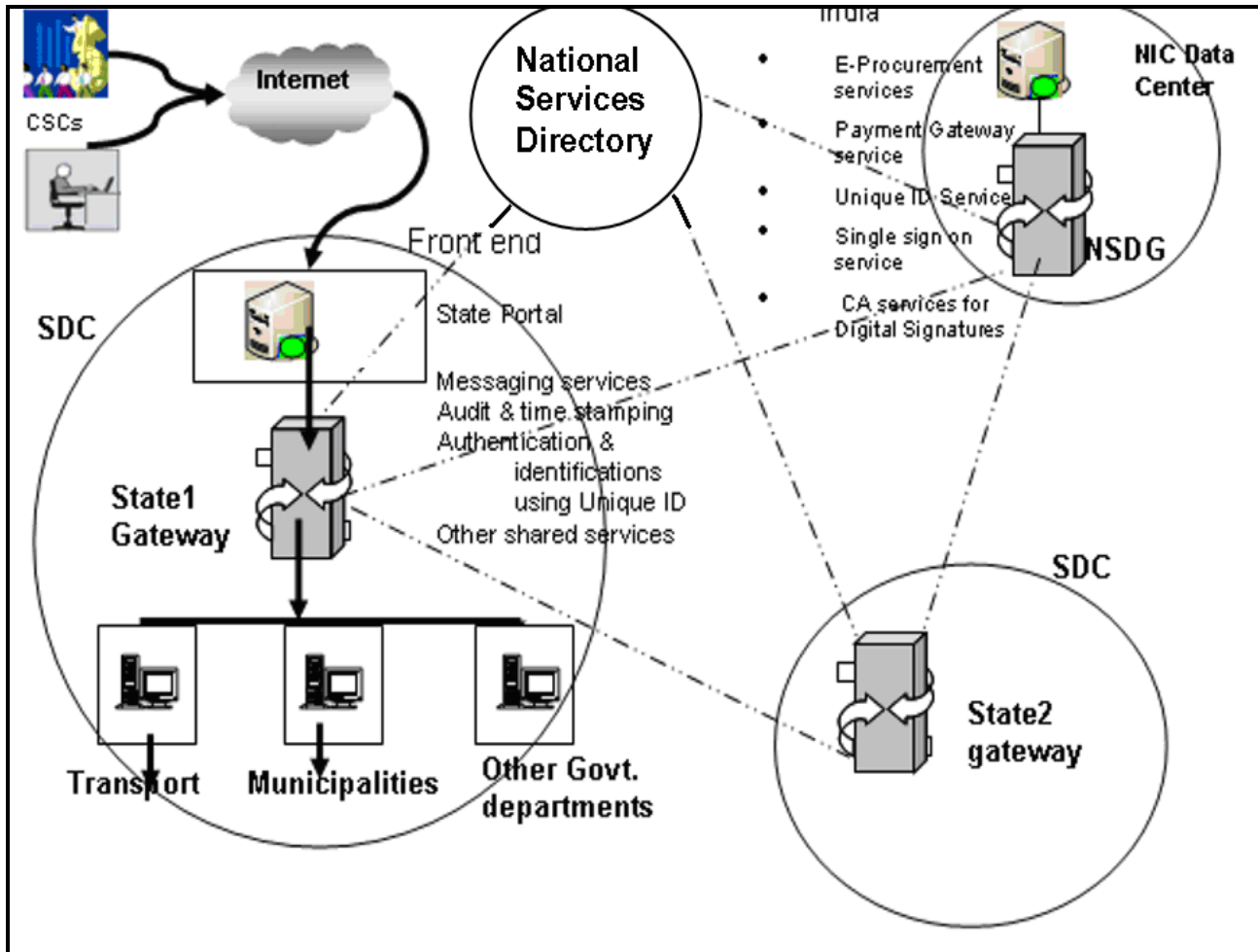


Figure 7 : Connectors in a Constellation of Gateways

In the above scenario for a constellation of Gateways, SAP and SP communicate in a similar pattern as discussed above. However, there is a possibility that the requested SP may not be registered on the same gateway that is Recipient Gateway as that of the SAP. In this case, when request from SAP reaches the gateway i.e. Service Hosting Gateway to which SP is registered, the gateway looks up the relevant Service Hosting Gateway in National Services Directory (NSD) and communicates the SAP request to the respective Service Hosting Gateway.

The working of connectors in a constellation of Gateways is explained below.

Conceptual Working of Connectors on SAP side

Request from SAP to avail SP service is sent through SSDG.

- a. SAP sends the request to SAP application specific connector.
- b. The message request sent by SAP application specific connector should be in the XML format as per the XSD (schema) defined by SP.
- c. Application specific connector uses the API i.e. class *ConnectorInterface* of the SAP generic connector to pass the request in XML format to SP side through SSDG.
- d. The generic connector converts the payload as per the IIP/IIS standards before sending it to the SSDG.
- e. The request being asynchronous, SSDG returns an acknowledgement for the same to SAP. The acknowledgement contains an ID (*correlationId* generated by SSDG) for the request. Using this ID, the citizen can track the status of his request by using poll operation on SSDG.
- f. SSDG (Gateway A in this case) looks up for the relevant SP registered with it to forward the request. If the SP is not registered with Gateway A, it will look up in the National Services Directory (NSD) to locate a gateway where the desired SP is registered. If found, the NSD returns the location of the gateway (Gateway B) where SP is registered.
- g. Gateway A will now pass the request to the Gateway B where SP is registered.
- h. Once the request reaches the Gateway B, it is channelized to SP as described below.

Conceptual Working of Connectors on SP side

The following business process is catered on SP side.

- a. The SP generic connector checks whether the request message sent by SSDG is valid and takes appropriate action in case of valid/invalid request messages.
- b. If the message sent from SSDG is invalid then the SP generic connector sends the message back to the Gateway B along with the error message conveying appropriate reason for the error.
- c. If the message sent is valid, the SP generic connector passes the message to the SP application specific connector by extracting the payload and other necessary details.
- d. Based on the payload and details provided by SP generic connector, SP application specific connector sends the request to SP whose service is requested.
- e. SP application specific connector then formats and sends back the response returned by SP application to SP generic connector using the SP generic connector API.(for asynchronous request, the response is returned using generic connector API whereas in case of synchronous request the response is sent in a single call to SP.
- f. SAP will get the response submitted by SP to Gateway B, SAP is supposed to poll the Recipient gateway i.e. Gateway A where it is registered. In first Poll Gateway A will return the address of Gateway B from where SAP can get response submitted by SP registered with Gateway B. In subsequent Poll to gateway B SAP will get the response submitted by Gateway B.

Note: It is noteworthy that in the above scenario where a constellation of Gateways is involved, when the Document Poll() is invoked for the first time, the request shall be submitted to the original Gateway. Original gateway i.e. Recipient Gateway will return the address of the Service Hosting Gateway where the SP is actually registered. SAP will invoke second document poll on Gateway B i.e. Service Hosting Gateway to get the response.

Note: As SAP request is available with two gateways i.e. Service Recipient Gateway where SAP is registered as well as Service Hosting Gateway where SP is registered, SAP is suppose to call delete request on both gateways to delete any SAP specific data available with both the gateways. It is responsibility of SAP to call delete request i.e. DocumentDelete() on Gateway A i.e. Service Recipient Gateway as it is only carrying the request submitted by the SAP as well as on service hosting gateway which contains request as well as response specific to SAP.

Part II: Technical Details

Chapter 3 Generic Connector

This Chapter covers all aspects of Generic Connector including SAP Generic Connector, SP Generic Connector, their properties and functionalities in detail. SAP Generic connector communicates with SAP application specific connector and SSDG. It provides APIs for SAP application specific connector to communicate with itself.

SP generic connector takes the request from SSDG and passes it to SP application connector by invoking appropriate functions on the SP application connector. The SP application connector finally calls the desired service of SP to serve the request.

The SP generic connector provides interface to the SP application connector using which SP application connector invokes a function *submitResponse()* to submit the response generated by SP application which is further submitted to SSDG.

Apart from assuring delivery and acknowledgement of requests between the SAP and SP, connectors also perform the indispensable task of authenticating the SAP to the SP so as to ensure only valid and authentic SAP interact with valid and authentic SP services. Before we go into the details about the types of connectors and their working, we will take a look at the authentication features provided by connectors.

3.1 Authentication in Connectors

When SAP makes a request to SP, SAP application connector ensures that the property pertaining to the Authentication Mode is set as per the requirement of service registered by SP. This Authentication Mode may be either of the following.

- i) **Clear Text** - If the desired Authentication Mode requested by SP is Clear Text in a request, then SAP application connector will set the property pertaining to Authentication Mode as *Clear Text*. SAP generic Connector will not encrypt the password set as Clear Text and pass the request to SSDG. As there is no encryption in Clear Text, the security ensured by this type of authentication mode is much lesser than SHA1 and Digital Signature.
- ii) **SHA1** - If the desired Authentication Mode by SP is SHA1, then SAP application connector will set the property pertaining to Authentication Mode as SHA1. SAP generic connector will perform SHA1 based hashing and base64 Encoding. The SHA1 encryption offers much more security and lesser vulnerability than Clear Text authentication mode.
- iii) **W3C Signed** – If the Authentication Mode requested by the SP is a digitally signed certificate in a request, then application connector will set the property pertaining to Authentication Mode as W3C Signed. It will also set the path of the file which is to be used for digital signature. SAP generic connector will then execute the algorithm for digital signing of the file before passing the request to SSDG. Digital signatures are the most secure way of ensuring information security than Clear Text or SHA1.

Note: Authentication Mode indicates how the SAP will be authenticated by SP. It is imperative that only authentic SAP interact with the SP. Hence every SP service dictates that any SAP interacting with it should validate itself through one of the above mentioned modes. The Authentication Mode is specified by the SP service and used by the gateway. Certificate Management pertaining to Digital Signature based Authentication mode will be done by Generic Connector but the path of certificates installed in the system needs to be specified in application configuration files for proper functioning of this mechanism

3.2 SAP Generic Connector

The SAP generic connector provides interface depending on the type (asynchronous/synchronous) of SAP request. This interface is exposed to the SAP application connector.

3.2.1 Responses that SAP receives

Three types of responses can be received on calling the functions;

Error- *SubmitRequest()* will return appropriate error if any of the inputs set by SAP application specific connector is invalid or gateway is unreachable.

Acknowledgement- A *CorelationID* with status acknowledging the receipt of request is returned in case of successful submission of request. Acknowledgements are returned only in case of Asynchronous Requests.

Response – The response for the submitted request

3.2.2 SAP Generic Connector Functions

There are 4 functions defined in SAP generic connector API implemented by the SAP application connector.

- a. *SubmitRequest()* - This function should be used for making initial request to SSDG for availing a service.
- b. *DocumentPoll()* – This function is used to check the status of the request which has been submitted through asynchronous mode.
- c. *DocumentDelete()* – This function is used to delete the request and response which have already been fulfilled successfully.
- d. *DocumentList()* – It helps in getting the status of all the requests submitted to SSDG during a particular period.

Each of these functions is explained in detail below in Section 3.1.3.

Note: Only *SubmitRequest()* is used for Synchronous as well as asynchronous Requests while all the other functions will use Synchronous based requests only.

3.2.3 SAP Generic Connector Properties

SAP generic connector properties which needs to set for making a request are given in *Table 1*.

Table 1 : SAP Generic Connector Properties

S.No	Property Name	Data Type	Description
1	SenderID	String	User Id to authenticate SAP in <i>SubmitRequest()</i> and <i>DocumentList()</i> methods only.
2	Password	String	Password to authenticate SAP Required in <i>SubmitRequest()</i> and <i>DocumentList()</i> methods only.
3	BodyContent	XMLElement[]	Information needed by SAP from SP should be set into this property and the content of this property will be as per SP Guidelines. This is applicable only for functions <i>SubmitRequest()</i> and <i>DocumentList()</i> .In <i>DocumentList()</i> , this property will contain the status of all requests made by SAP and content of this should be as per schema mentioned in Appendix D .
4	CorrelationID	String	Required for functions <i>Document Poll()</i> and <i>DocumentDelete()</i> where SAP gets this ID in the first submission to SSDG for asynchronous types of request, whereas there is no option for <i>DocumentPoll()</i> and <i>DocumentDelete()</i> for synchronous type of request.
7	ResponseMode	Enum	Response mode specifies the type of communication needed by SP. It depends upon the SP service in which mode the SP provides its services i.e. asynchronous or synchronous.
8	ClassValue	String	End point of service requested by the SAP, it will be provided to SAP in its registration details.
9	TargetEndpoint	String	Target Point of the SSDG which will be used by SAP and SP.
10	SubmissionStatus	String	Submission status given to SAP from SSDG through Generic connector. encryption algorithm applied to it.

11	ErrorList	String []	If submission fails at SSDG due to any problem at SSDG, it will be notified to SAP through ErrorList.
12	Error Code	String	Number specific to error condition, for error codes description, refer Appendix A .
13	SignedAuthentication	Enum	If the requested service needs an XMLSignature level authentication, this field needs to be set by SAP and Certificate details related to Certificate management needs to be specified in the configuration file or at the server where the generic connector is getting deployed. Applicable for method <i>SubmitRequest and List Request</i>) only.

3.2.4 SAP Generic Connector Functions

The SAP Generic Connector Functions defined in the Generic Connector API are explained in detail below with their sequence diagrams. The inputs and outputs for the various request modes and authentication types are also provided with explanation.

3.2.4.1 Sequence for Asynchronous Submit Request

The sequence diagram in Fig.8 shows the illustrative flow of **Asynchronous Request** from SAP to SP. It also depicts how SP routes the *Response* back to SAP. During this exchange of *Request* and *Response*, several function calls are made between SAP and SP connectors. This handshake between various components involved is elaborated ahead.

It is worthwhile noting here that since the *Request* call is *Asynchronous*, the citizen does not receive the *Response* immediately. Alternatively, the citizen will receive an acknowledgement implying the receipt of the *Request*. The *Response* is generated and delivered at a later stage.

The following steps describe how SAP and SP interact with each other through the SSDG by invoking various function calls.

- 1) Suppose a citizen wants to apply for his/her Birth Certificate through relevant Govt. department. The citizen initiates a Request for the same using SAP application.(Cycle 1)
- 2) SAP puts the request in SP desired format and forwards it to SAP side Application Specific Connector.

- 3) This Request is then passed to SAP Generic Connector by invoking a function *SubmitRequest()* on SAP Generic Connector.
- 4) SAP Generic Connector delivers this Request to SSDG. SSDG then sends an acknowledgement along with a unique CorrelationID to SAP Generic Connector. Generic Connector passes it back to SAP Application Specific Connector which delivers it to SAP Application from where it finally reaches to the citizen. This CorrelationID helps the citizen in tracking the status of the Request submitted. Thus, Cycle 1 as depicted in Fig.8 is completed and the connection with SSDG is broken.
- 5) Now, Cycle 2 as shown in Fig.8 begins and SSDG delivers the Request to SP Generic Connector.
- 6) This Request is further forwarded to SP Application Specific Connector by invoking the function *asyncSubmitRequest()*.
- 7) SP Application Specific Connector calls the appropriate service on SP and passes the Request. The acknowledgement for the same is conveyed to SSDG through SP Specific and Generic connectors respectively.

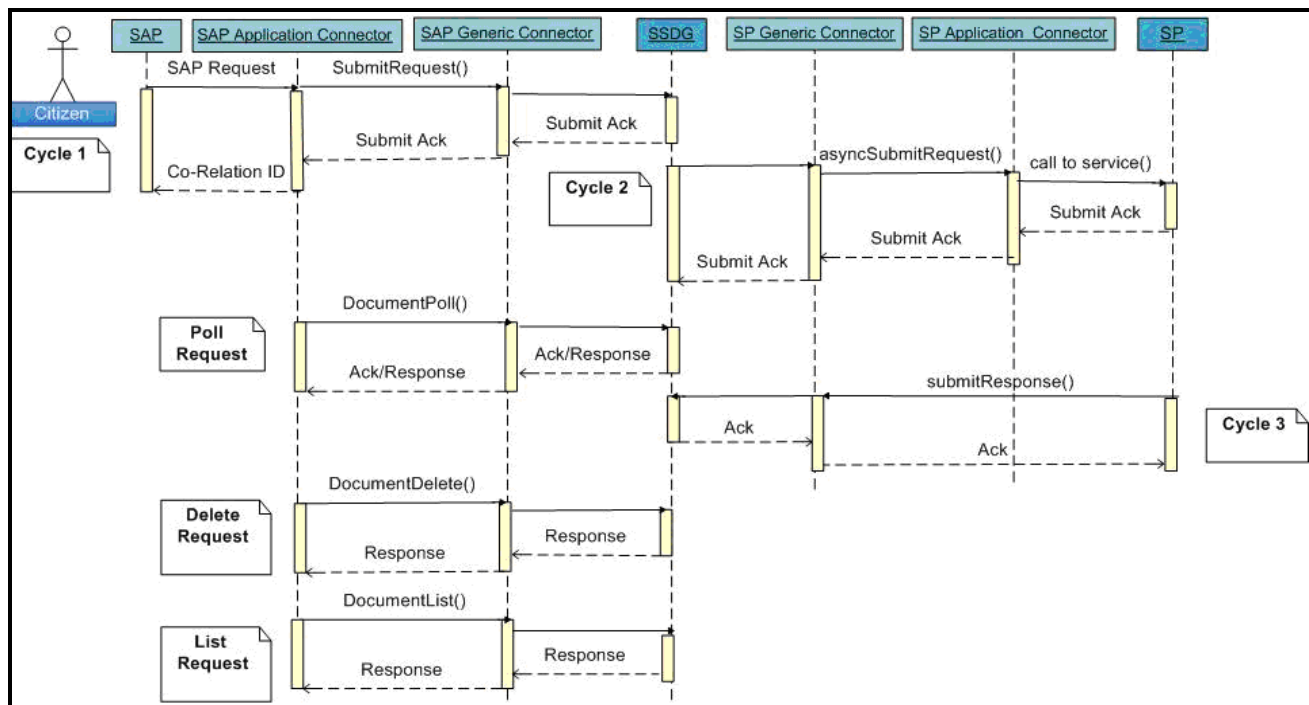


Figure 8: Sequence Diagram for Asynchronous Request

- 8) Now, as Cycle 3 in Fig.8 depicts, SP will generate the appropriate Response once the Request has been successfully processed and send Response to SP Application Specific Connector.

- 9) SP Application Specific Connector invokes the *submitResponse()* function on SP Generic Connector and pass the Response to it.
- 10) SP Generic Connector will then forward this Response to SSDG. SSDG will acknowledge the receipt of the Response to SP. Thus, Cycle 3 gets completed on submission of Response to SSDG.

Poll Request

After submitting the Request and receiving the CorrelationID, the citizen can check the status of his Request. SAP caters to this request by calling *DocumentPoll()* function.

Poll Request in Fig.8 depicts this execution.

- 1) The citizen checks the status of his/her Request by submitting the CorrelationID to SAP application.
- 2) This Request is picked up by SAP Application Specific Connector which invokes the *DocumentPoll()* function on SAP Generic Connector.
- 3) SAP Generic Connector gets status of the Request from SSDG. If the response or error has already been submitted by SP, the same is returned in the form of Response to citizen else an acknowledgement is sent until response is awaited from SP end.

Delete Request

Additionally, if the citizen's Request has been fulfilled and he/she wishes to remove the Request, he/she can do so by submitting the corresponding CorrelationID for the Request.

Delete Request in Fig.8 depicts this execution.

- 1) The citizen submits the CorrelationID pertaining to the request to be deleted to SAP application.
- 2) This Request is picked up by SAP Application Specific Connector which invokes the *DocumentDelete()* function on SAP Generic Connector.
- 3) SAP Generic Connector passes the Delete Request to SSDG which in turn will delete the corresponding Request. It then generates a Response acknowledging the delete action and returns the same to the Citizen.

Note: Note that in the case of constellation of gateways to delete a request completely DocumentDelete() needs to be called twice, once to remove it from the domain gateway and then to remove it from the SSDG. As SAP request is available with two gateways i.e. Service Recipient Gateway where SAP is registered as well as Service Hosting Gateway where SP is registered, SAP is suppose to call delete request on both gateways to delete any SAP specific data available with both the gateways. It is responsibility of SAP to call delete request i.e. DocumentDelete() on Gateway A i.e. Service Recipient Gateway as it is only carrying the request submitted by the SAP as well as on service hosting gateway which contains request as well as response specific to SAP.

List Request

The citizen can also view all *Requests* posted for a particular time interval.

List Request in Fig.8 depicts this execution.

- 1) When a citizen specifies a particular interval for which he/she wishes to list the requests in the SAP Application, this *Request* is picked up by *SAP Application Specific Connector* which invokes the *DocumentList()* function on *SAP Generic Connector*.
- 2) SAP Generic Connector passes the *List Request* to SSDG which will respond by listing all Requests pertaining to the period specified by the citizen.

Details of API functions

a) Asynchronous Submit Request

- i) Type of Response Mode- **Asynchronous**
Authentication Type- **Clear**

Input

Table 2 : Input Parameters for Asynchronous Submit Request

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider URL	Mandatory
2	BodyContent	Content of information needed by SP to process the SAP request.	Mandatory
3	AuthMode	Clear	Mandatory
4	SenderID	SenderID provided at the time of registration	Mandatory
5	Password	Password provided at the time of registration	Mandatory
6	TargetEndpoint	URL of SSDG	Mandatory
7	ResponseMode	Asynchronous	Mandatory
8	TransactionID	Optional field set by SAP.	Optional

Output

Table 3 : Output Parameters for AsynchronousSubmitRequest in case of Response

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Acknowledgement or Error
5	Error List	If Submission Status is error, then its details.
6	Error Code	Error code specifying exact error.

- ii) Type of Response Mode- **Asynchronous**
Authentication Type- **SHA1**

Input

Table 4 : Input Parameters for Asynchronous Submit Request with SHA1 encoding

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	AuthMode	SHA1	Mandatory
4	SenderID	SenderID provided at the time of registration	Mandatory
5	Password	Password provided at the time of registration	Mandatory
6	TargetEndpoint	URL of SSDG	Mandatory
7	ResponseMode	Asynchronous	Mandatory
8	TransactionID	Optional field set by SAP.	Optional

Output

Table 5 : Output Parameters for AsynchronousSubmitRequest in case of Response

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 bit Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Acknowledgement or Error
5	Error List	If Submission Status is error, then its details.
6	Error Code	Error code specifying exact error.

- iii) Type of Response Mode- **Asynchronous**
Authentication Type- **W3CSigned**

Input

Table 6 : Input Parameters for Asynchronous Submit Request with digital signature

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	AuthMode	W3CSigned	Mandatory
4	SenderID	SenderID provided at the time of registration	Mandatory
5	Password	Password provided at the time of registration	Not Required for W3CSigned mode of authentication.
6	TargetEndpoint	URL of SSDG	Mandatory
7	ResponseMode	Asynchronous	Mandatory
8	TransactionID	Optional field set by SAP.	Optional
9	Certificate Information	Configuration file or Deployment Server	Mandatory

Output

Table 7 : Output Parameters for Asynchronous Submit Request in case of Response

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Acknowledgement or Error
5	Error List	If Submission Status is error, then its details.
6	Error Code	Error code specifying exact error.

b) Document Poll

To get the response of previously submitted request SAP polls SSDG Gateway for the requests which SAP submitted in an asynchronous mode.

Input

Table 8 : Document Poll Input Parameters

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL where	Mandatory

		asynchronous <i>SubmitRequest()</i> has been made.	
2	TargetEndpoint	URL of SSDG	Mandatory
3	TransactionID	Optional field set by SAP.	Optional
4	CorrelationID	Received after asynchronous <i>SubmitRequest()</i>	Mandatory

Output

Table 9 : Document Poll Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (optional)
4	Submission Status	Acknowledgement/Response/Error
5	Error List	If Submission Status is error, lists details
6	Body Content	If Submission Status is response.
7	Error Code	Error code specifying exact error if error is received.

c) Document Delete

Delete previously submitted request and response from the SSDG Gateway for asynchronous type of requests.

Input

Table 10 : Document Delete Input Parameters

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL where asynchronous <i>SubmitRequest()</i> have been made.	Mandatory
2	TargetEndpoint	URL of SSDG	Mandatory
3	TransactionID	Optional field set by SAP.	Optional
4	CorrelationID	Received after asynchronous <i>SubmitRequest()</i>	Mandatory

Output

Table 11 : Document Delete Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Response or Error
5	Error List	If Submission Status is error, lists details.

6	Error Code	Error code specifying exact error if error is received.
---	------------	---

d) Document List

Lists the status of all the records submitted to SSDG for a particular service and mode with relevant authentication details.

- i) Type of Response Mode – **Asynchronous / Synchronous**
Authentication Type- **Clear**

Input

Table 12 : Document List Input Parameters

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	AuthMode	Clear	Mandatory
3	SenderID	SenderID provided at the time of registration	Mandatory
4	Password	Password provided at the time of registration	Mandatory
5	TargetEndpoint	URL of SSDG	Mandatory
6	ResponseMode	Asynchronous/Synchronous	Mandatory
7	TransactionID	Optional field set by SAP.	Optional
8	BodyContent	Request as per request schema for List request in Appendix D.	Mandatory

Output

Table 13 : Document List Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Response or Error
5	Error List	If Submission Status is error, lists details.
6	BodyContent	Response/Error from SSDG as per response schema for List request in Appendix D.
7	Error Code	Error code specifying exact error if error is received.

- ii) Type of Response Mode – **Asynchronous / Synchronous**
Authentication Type- **SHA-1**

Input

Table 14 : Document List Input Parameters with SHA1 encoding



S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	AuthMode	SHA1	Mandatory
3	SenderID	SenderID provided at the time of registration	Mandatory
4	Password	Password provided at the time of registration	Mandatory
5	TargetEndpoint	URL of SSDG	Mandatory
6	ResponseMode	Asynchronous/Synchronous	Mandatory
7	TransactionID	Optional field set by SAP.	Optional
8	BodyContent	Request as per request schema for List request in Appendix D .	Mandatory

Output

Table 15 : Document List Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider (SP)URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Response/Error
5	Error List	If Submission Status is error, lists details.
6	BodyContent	Response/Error from SSDG as per response schema for List request in Appendix D .
7	Error Code	Error code specifying exact error if error is received.

iii) Type of Response Mode – **Asynchronous / Synchronous** Authentication Type- **W3CSigned**

Input

Table 16 : Document List Input Parameters with Digital Signature

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider URL	Mandatory
2	AuthMode	W3CSigned	Mandatory
3	SenderID	SenderID provided at the time of registration	Mandatory
4	Password	Password provided at the time of registration	Not Required for W3CSigned mode of authentication.
5	TargetEndpoint	URL of SSDG	Mandatory
6	ResponseMode	Asynchronous/Synchronous	Mandatory
7	TransactionID	Optional field set by SAP.	Optional
8	Certificate Information	Configuration file or Deployment Server	Mandatory
9	BodyContent	Request as per request schema for List	Mandatory

request in **Appendix D**.

Output

Table 17 : Document List Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider (SP)URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	If provided by SAP
4	Submission Status	Response Error
5	Error List	If Submission Status is error.
6	Error Code	Error code specifying exact error if error is received.
7	BodyContent	Response/Error from SSDG as per response schema for List request in Appendix D .

3.2.4.2 Sequence for Synchronous Submit Request

The Sequence Diagram in Fig.9 shows the illustrative flow of *Synchronous Request* from SAP to SP. It also depicts how SP routes the *Response* back to SAP. During this exchange of *Request* and *Response*, several function calls are made between SAP and SP connectors. This handshake between various components involved is elaborated ahead.

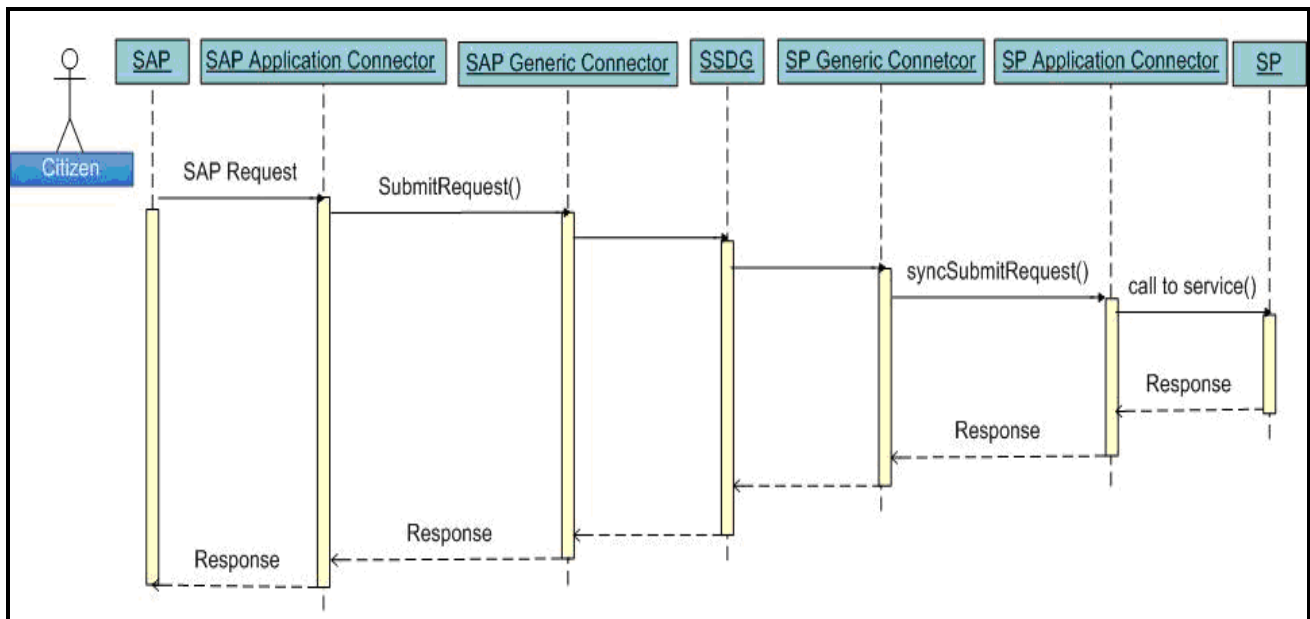


Figure 9 : Sequence Diagram for Synchronous Request

The following steps describe how SAP and SP interact with each other through SSDG by invoking various function calls.

- 1) Supposing a citizen wants to look up for details pertaining to a telephone number. The citizen initiates a *Request* for the same using SAP application.
- 2) SAP converts this Request into the format that is understood by SP and routes it to SAP Application Specific Connector.
- 3) This Request is then passed to SAP Generic Connector by invoking a function *SubmitRequest()* on SAP Generic Connector.
- 4) SAP Generic Connector delivers the Request to SSDG which in turn delivers it to SP Generic Connector.
- 5) SP Generic Connector invokes the *synSubmitRequest()* on SP Application Specific Connector and forwards the Request to SP Application Specific Connector.
- 6) SP Application Specific Connector then routes it to SP through appropriate service call and awaits Response from SP.
- 7) As this is a Synchronous call, SP processes the desired Request and delivers the Response to SP Application Specific Connector without breaking the connection.
- 8) SP Application Specific Connector then routes the Response back to SP Generic Connector from where it is passed to SSDG which channels it back to SAP Generic Connector. The Response finally reaches SAP Application through SAP Application Connector and is delivered to the citizen.

Details of the API Functions

a) Synchronous Submit Request

- i) Type of Response Mode – **Synchronous**
Authentication Type- **Clear**

Input

Table 18 : Input Parameters for Synchronous Submit Request

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	AuthMode	Clear	Mandatory
4	SenderID	SenderID provided at the time of registration	Mandatory
5	Password	Password provided at the time of registration	Mandatory

S.No	Property Name	Values	Requirement
6	TargetEndpoint	URL of SSDG	Mandatory
7	ResponseMode	Synchronous	Mandatory
8	TransactionID	Optional field set by SAP.	Optional

Output

Table 19 : Output Parameters for Synchronous Submit Request in case of Response

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Response or Error
5	Error List	If Submission Status is error, then its details.
6	BodyContent	Blank/ Response/ Error from SP.
7	Error Code	Error code specifying exact error if error is received.

- ii) Type of Response Mode - **Synchronous**
Authentication Type- **SHA1**

Input

Table 20 : Input Parameters for Synchronous Submit Request with SHA1 encoding

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	AuthMode	SHA1	Mandatory
4	SenderID	SenderID provided at the time of registration	Mandatory
5	Password	Password provided at the time of registration	Mandatory
6	TargetEndpoint	URL of SSDG	Mandatory
7	ResponseMode	Synchronous	Mandatory
8	TransactionID	Optional field set by SAP.	Optional

Output

Table 21 : Output Parameters for Synchronous Submit Request in case of Response

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Response or Error
5	Error List	If Submission Status is error, lists the details.

S.No	Property Name	Values
6	BodyContent	Blank/Response/Error from SP.
7	Error Code	Error code specifying exact error if error is received

iii) Type of Response Mode – **Synchronous**
Authentication Type- **W3CSigned**

Input

Table 22 : Input Parameters for Synchronous Submit Request with Digital Signature

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	AuthMode	W3CSigned	Mandatory
4	SenderID	SenderID provided at the time of registration	Mandatory
5	Password	Password provided at the time of registration	Not Required for W3CSigned mode of authentication.
6	TargetEndpoint	URL of SSDG	Mandatory
7	ResponseMode	Synchronous	Mandatory
8	TransactionID	Optional field set by SAP.	Optional
9	Certificate Information	Configuration file or Deployment Server	Mandatory

Output

Table 23 : Output Parameters for Synchronous Submit Request in case of Response

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (Optional)
4	Submission Status	Response or Error
5	Error List	If Submission Status is error, lists the details.
6	BodyContent	Blank/Response/Error from SP.
7	Error Code	Error code specifying exact error if error is received.

3.3 SP Generic Connector

The SP generic connector receives the request message destined for SP from SSDG in synchronous / asynchronous mode and responds accordingly. It invokes the *synSubmitRequest()* / *asynSubmitRequest()* of SP application connector for synchronous and asynchronous calls respectively for submitting the request to the SP application connector. The SP application connector will call the appropriate service of SP to cater to the request.

The SP generic connector also carries out end-to-end integrity checks to ensure that the request message has not been modified or tampered with during transit.

When response for asynchronous request is generated by SP application, the SP application connector submits this response to SP generic connector by invoking *submitResposne()* defined in SP generic connector API. This response is then submitted to SSDG from where it is served to the citizen through poll request.

3.3.1 SP Generic Connector Functions

The SP generic connector API defines a single function as mentioned below;

submitResposne() - The SP application connector invokes this function on SP generic connector using the API provided by SP generic connector. The function *submitResponse()* of class *SPImpl* will be used to submit the response using the properties defined in Table 24 to Generic connector. The generic connector then forwards it to SSDG.

This function is explained in detail below in Section 3.3.2

3.3.2 SP Generic Connector Properties

SP generic connector properties are given in Table 24.

Table 24 : SP Generic Connector Properties

S.No	Property Name	Data Type	Description
1	BodyContent	XMLElement[]	Information needed by SAP from SP should be set into this property and the content of this property will be as per SP Guidelines.
2	CorrelationID	String	CorrelationID set by SSDG.
3	TransactionID	String	Optional Element for all methods of Generic Connector. Can be used by SAP and SP application to keep the track of their document submission.
4	ResponseMode	Enum	Response mode specifies the type of

S.No	Property Name	Data Type	Description
			communication needed by SAP. It depends upon the SP service in which mode the SP provides its services i.e. asynchronous or synchronous.
5	ClassValue	String	End point of service requested by the SAP, it will be provided to SAP in its registration details.
6	TargetEndpoint	String	Target Point of the SSDG which is will be used by SAP and SP.
7	SubmissionStatus	String	Submission status given to SAP from SSDG through Generic connector.
8	ErrorList	String []	If submission gets failed at SSDG due to any problem at SSDG, it will be notified to SAP through generic connector.
9	Error Code	String	Number specific to error condition, for error codes description please refer Appendix A .

3.3.3 Details of SP Generic Connector Functions

The *SP Generic Connector function submitResponse()* defined in the Generic Connector API is explained in detail below along with the inputs and outputs for asynchronous mode.

a) *submitResponse()*

Type of Response Mode-Asynchronous

Input

Table 25 : Submit Response Input Parameters

S.No	Property Name	Values	Required
1	ClassValue	Service Provider URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	TargetEndpoint	URL of SSDG	Mandatory
4	ResponseMode	Asynchronous	Mandatory
5	TransactionID	Optional field set by SAP.	Optional
6	CorrelationID	32 Hexadecimal ID	Mandatory
7	Submission Status	“response” or “error”. SP should specify this for Generic connector to handle the request properly. Error is basically pertaining to business related problems and body content for it should be as per schema mentioned in Appendix D .	Mandatory

Output

Table 26 : Submit Response Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	If provided by SAP
4	Submission Status	Acknowledgement/Error
5	Error List	If Submission Status is error.
6	Error Code	Error Details

Chapter 4 Application Connector

An application specific connector uses the APIs exposed by the generic connectors to communicate with SSDG. Application specific connector also converts the message in a format that is understandable to SP. They are responsible for service resolution. Also, at SP side, the SP application specific connector identifies the methods of the SP application registered against the *Class Ids*. These methods are responsible for processing the request and generating response for the same which is then catered to SAP.

4.1 SAP Application Connector

The SAP application specific connector accepts the request from an SAP application and converts the request message in XML format for compliance with format specified by SP for availing its services. It uses the interface exposed by the SAP generic connector and assigns the properties for the request. Finally, it invokes appropriate functions on the SAP generic connector and passes the request to it. The request is then submitted to SSDG by the SAP generic connector.

Technical Note: Please include the App.Config file in your application with SAP Generic Connector. The App.Config file contains properties and values and that are not permanent in nature and may change over time or as per the discretion of SAP application developer. The App.Config file is given in Appendix B.

4.2 SP Application Connector

SP application specific connector implements the methods defined in the interface *IServiceSoap* provided in the form of a DLL. SP application specific connector implements the methods *synSubmitRequest()* and *asynSubmitRequest()* in the form of a Web Service. These methods are called by the SP generic connector for submitting asynchronous/ synchronous requests received from SSDG for targeted SP Application. The developer needs to build SP application specific connector in the form of a Web Service and will use the properties defined in the provided DLL. Description of these properties is given in Table 24.

SP application specific connector will also define a method for submitting a response to SSDG

through SP generic connector. This method will receive the values in the parameters as listed in Table 27. The SP application processes the request as per the application logic and after processing, the response/ error as per the content processed is sent back in response by invoking *submitResponse()* on SP generic connector.

Technical Note: Please include the Web.Config file in your application with SP Generic Connector. The Web.Config file contains properties and values and that are not permanent in nature and may change over time or as per the discretion of SP developer. The Web.Config file is given in Appendix C.

4.2.1 SP Application Connector Functions

The SP application connector interface defines the following functions;

- a) *synSubmitRequest()*
- b) *asynSubmitRequest()*

a) Asynchronous Submit Request

- i) Type of Response Mode – **Asynchronous**

Input

Table 27: Asynchronous Submit Request Input Parameters

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	TargetEndpoint	URL of SSDG	Mandatory
4	ResponseMode	Asynchronous	Mandatory
5	TransactionID	Optional field set by SAP.	Optional
6	CorrelationID	32 Hexadecimal ID	Mandatory

b) Synchronous Submit Request

- i) Type of Response Mode – Synchronous

Input

Table 28 : Synchronous Submit Request Input Parameters

S.No	Property Name	Values	Requirement
1	ClassValue	Service Provider (SP) URL	Mandatory
2	BodyContent	Content of information needed from SP	Mandatory
3	TargetEndpoint	URL of SSDG	Mandatory

4	ResponseMode	Synchronous	Mandatory
5	TransactionID	Optional field set by SAP.	Optional
6	CorrelationID	32 Hexadecimal ID	Mandatory

Output

Table 29 : Synchronous Submit Request Output Parameters

S.No	Property Name	Values
1	ClassValue	Service Provider (SP) URL
2	CorrelationID	32 Hexadecimal ID
3	TransactionID	ID provided by SAP (optional)
4	BodyContent	Response/Error from SP
5	Submission Status	“response” or “error”. SP should specify this for Generic connector to handle the request properly. Error is basically pertaining to business related problems and body content for it should be as per schema mentioned in Appendix D .

Part III: Connector Troubleshooting

Chapter 5 Testing and Troubleshooting

This section deals with troubleshooting errors which are likely to arise and cause disruption in the smooth functioning of connectors. Below is a guide that points out the input errors and improper property settings in connector methods, their implications and the correct input and output values.

5.1 SAP Connector

i) Submit Request

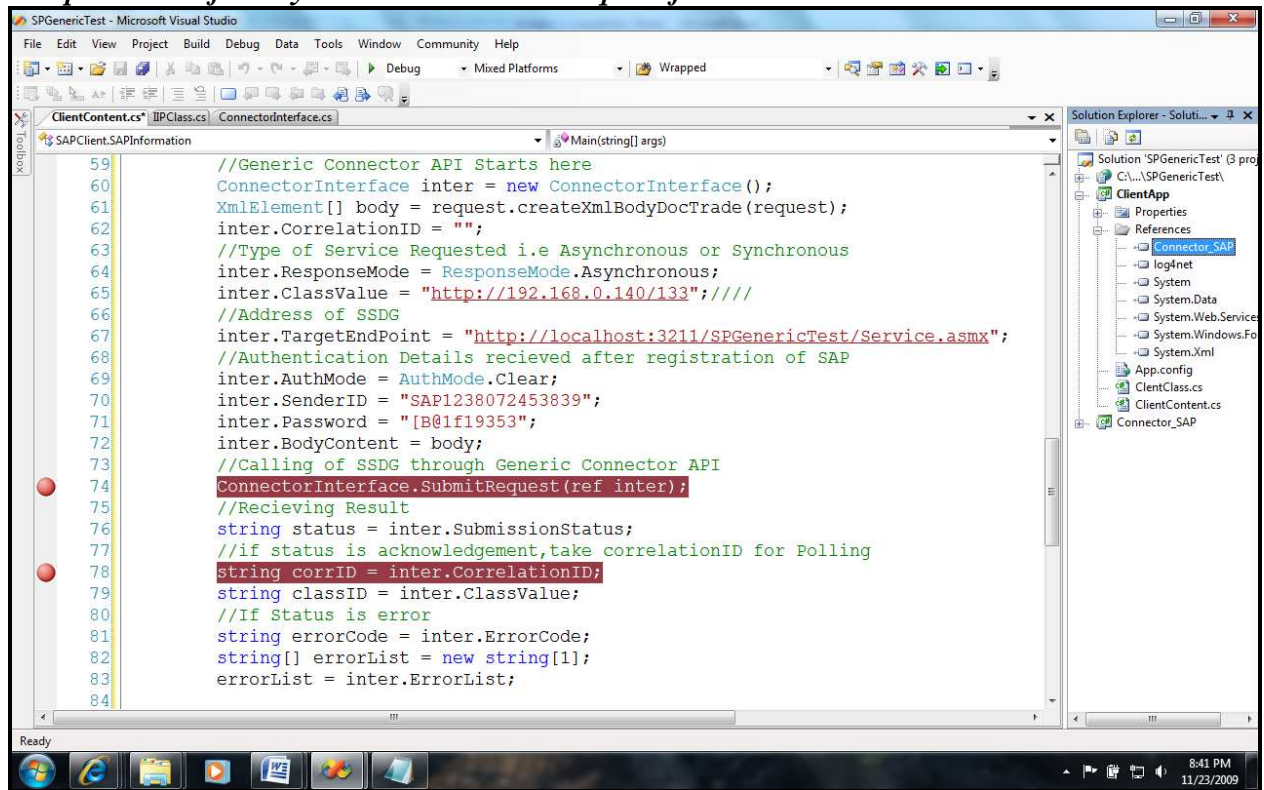
Input-Correct way of specifying values in Submit Request

The following lines of code show the correct property settings while invoking the SubmitRequest() on SAP generic connector. All properties pertaining to SubmitRequest() are explained in Table 1 . The comments provide the description of the properties.

```
//Set Values in the Object of SP Generic API provided
ConnectorInterface inter = new ConnectorInterface ( );
//Correct way of specifying the values for transaction ID
inter.TransactionId = "12345678912345678912345678912345";
//or
inter.TransactionId = ""; //Will give response as it is optional field
//Corrlation ID should be left blank for Submit Request
inter.CorrelationID = "";
//Response Mode of the service as for which SAP is registered.
inter.ResponseMode = ResponseMode.Asynchronous; //Asynchronous Mode
//or
inter.ResponseMode = ResponseMode.Synchronous; //Synchronous Mode
//End Point URL for the SP service provided at the time of Service registration by SAP.
inter.ClassValue = "http://127.0.0.1/22";
//Depending upon the type of Authentication used by Service.
inter.AuthMode = AuthMode.Clear;
//or
inter.AuthMode = AuthMode.SHA1;
//or
inter.AuthMode = AuthMode.W3CSigned;
//If W3CSigned provide Keystore path in Configuration file i.e App.config file for certificates used for signing
the document.
//SenderID for Authentication Purpose
inter.SenderID = "SAP130720081454";
//Password for Authentication Purpose.Only applicable for clear and hash types of authentication.
inter.Password = "[B@13a87a0";
//End Point Url of the Gateway.
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
//XML Content in the form of XMLElement[].Client specific content.
inter.BodyContent = body;
//Final Submission to SPGeneric Connector.
submitRequest method is getting called.
```

```
Inter.SubmitRequest ( ref inter );  
//Apart from the above values also specify correct entries in App.config file available in Appendix B
```

Sample Screen for Asynchronous Submit Request for clear mode



Output-Response for Submit Request(Asynchronous Mode)

The following is the output of SubmitRequest() in Asynchronous mode. If the property settings are correct, the output status will be an acknowledgement for the SubmitRequest().

```
//Response for Correct Asynchronous Submit Request and succesful submission at Gateway.
```

```
//As the parameter is passing by ref ,the values for output can be retrived from the same object which is passed as parameter with all input values.
```

```
//Status of submission
```

```
//Output Values
```

```
// if the submission is in Asynchronous mode
```

```
inter.SubmissionStatus = "Acknowledgement";
```

```
//Valid 32 Bit value provided by gateway.Should be given this ID in documentPoll and documentDelete methods.
```

```
inter.CorrelationID = "12345678912345678912345678912345";
```

```
//If provided by the SAP.It remains Unchanged from Gateway.
```

```
inter.TransactionId = "12345678912345678912345678912345";
```

```
//End Point Url of the Gateway where documentPoll,documentDelete and listRequest can be done
```

```
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
```

Output-Response for Submit Request(Synchronous Mode)

The following is the output of SubmitRequest() in Synchronous mode. If the property settings are correct, the output status will be an acknowledgement for the SubmitRequest().

```
inter.SubmissionStatus = "response";  
//If response take the response given by SP in xmlelement[].  
inter.BodyContent = body;  
//documentPoll and documentDelete is not valid for Synchronous Submit Request  
inter.CorrelationID = "12345678912345678912345678912345";  
//If provided by the client.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";
```

Input-Incorrect Way of specifying values in Submit Request

The following lines of code show the incorrect property settings while invoking the SubmitRequest() on SAP generic connector. The comments provide the reason for the errors that arise due to incorrect property settings.

```
//InCorrect way of specifying the values  
//Error as the ID is not 32 bit hexadecimal.  
inter.TransactionId = "1234444";  
//Will give an error as this should be left blank;  
inter.CorrelationID = "12345678912345678912345678912345";  
//Will give an error as value is blank,Should provide correct value for class.  
inter.ClassValue = "";  
//Will Give an error,should provide correct value for senderID.  
inter.SenderID = "";  
//Will Give an error,should provide correct value for password  
inter.Password = "";  
//Will Give an error,should provide correct value for body in the form of XMLElement[].  
inter.BodyContent = body;  
//End Point Url of the Gateway  
inter.TargetEndPoint = "http://localhost/SPService;
```

Output-Response for Submit Request(Synchronous/Asynchronous Mode)

The following is the output of SubmitRequest() in Synchronous/Asynchronous mode in case of incorrect property settings. The output status will generate an error with an error code and error list. The explanation for the error code can be looked up in the error codes list specified in Appendix A or in the ErrorList given in the output.

```
//As the parameter is passing by ref ,the values for output can be retrived from the same object which is passed  
as parameter with all input values.  
//Status of submission  
//Output Values  
//if submission fails at Gateway
```

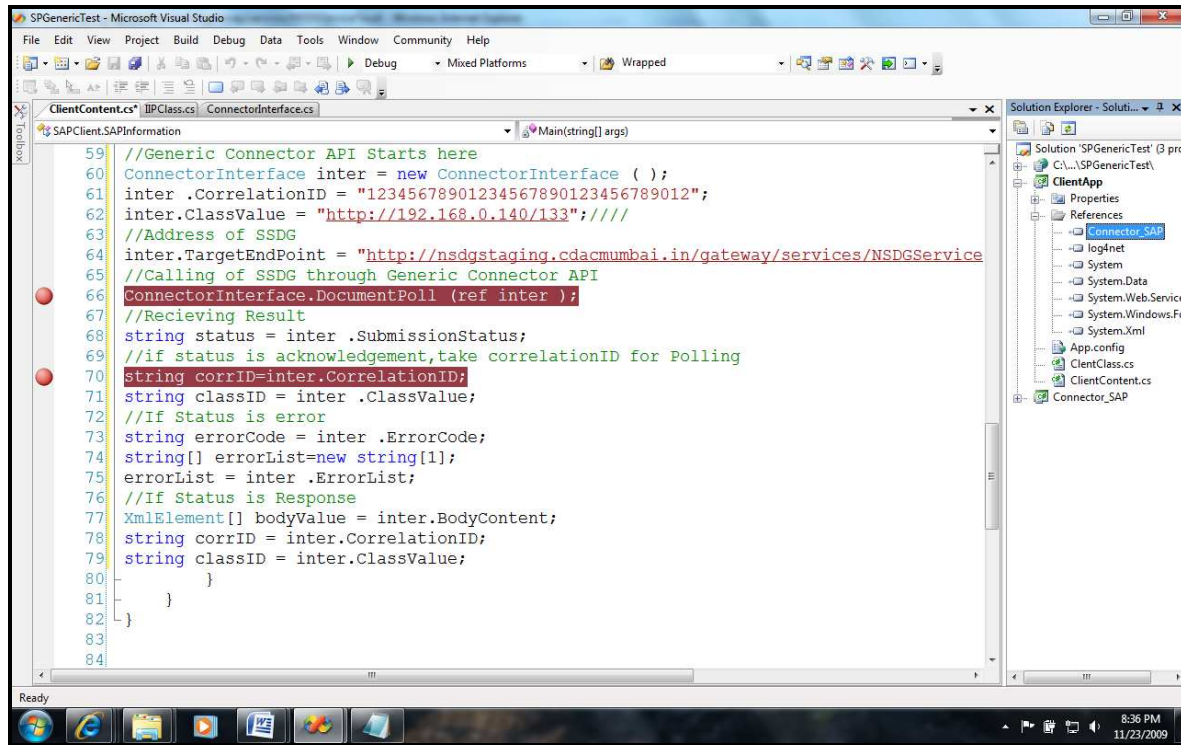
```
inter.SubmissionStatus = "error";  
//Error code detailed description related to error code can be viewed using the Table given in Appendix A or  
retriving the error description in the errorList field.  
inter.ErrorCode = "1033";  
//Reason for the error.  
inter.ErrorList = "Error specifying the reason";  
//If provided by the client.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";
```

ii) DocumentPoll

Input-Correct way of specifying values in Document Poll

The following lines of code show the correct property settings while invoking the DocumentPoll() on SAP generic connector. All properties pertaining to DocumentPoll() are explained in Table 8. The comments provide the description of the properties.

```
//Set Values in the Object of API provided  
ConnectorInterface inter = new ConnectorInterface ();  
inter.TransactionId = "12345678912345678912345678912345";  
//or  
//Will give response as it is optional field  
inter.TransactionId = "";  
//It should contain the value as received in Asynchronous SubmitRequest  
inter.CorrelationID = "12345678912345678912345678912345";  
//End Point URL for the service which is used in Asynchronous SubmitRequest  
inter.ClassValue = "http://127.0.0.1/22";  
//End Point Url of the Gateway where documentPoll can be done  
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";  
//Final Submission to SPGeneric Connector.  
submitRequest method is getting called.  
inter.documentPoll ( ref inter );  
//Apart from the above values specify correct entries in App.config file also.
```



Output-Response for Document Poll

The following is the output of DocumentPoll(). If the property settings are correct, the output status will be an acknowledgement for DocumentPoll()

```
//Response for Correct document poll and succesful submission at Gateway.
//As the parameter is passing by ref ,the values for output can be
retrived from the
//same object which is passed as parameter with all input values.
//Status of submission
//Output Value
// if the submission status is Acknowledgement,it implies that SP still not submitted the resposne.
inter.SubmissionStatus = "Acknowledgement";
inter.CorrelationID = "12345678912345678912345678912345";
//If provided by the client.It remains Unchanged from Gateway.
inter.TransactionId = "12345678912345678912345678912345";
//End Point Url of the Gateway
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
//if the submission status is response,it implies that SP has submitted the response.
inter.SubmissionStatus = "response";
//take the response given by SP in xmlelement[].
inter.BodyContent = body;
```



```
//Valid 32 Bit value provided by gateway.Should be given this ID in documentDelete methods.
inter.CorrelationID = "12345678912345678912345678912345";
//If provided by the client.It remains Unchanged from Gateway.
```

```
inter.TransactionId = "12345678912345678912345678912345";
//End Point Url of the Gateway
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
inter.SubmissionStatus = "error";
//Check the error code in ErrorCode field.If error code is related to business error then retrieve the content in
Body as per Error Schema mentioned in Appendix.
inter.ErrorCode = "1033";
//Reason for the error.
inter.ErrorList = "Error specifying the reason";
inter.BodyContent = body;
//Valid 32 Bit value provided by gateway.Should be given this ID in documentDelete methods.
inter.CorrelationID = "12345678912345678912345678912345";
//If provided by the client.It remains Unchanged from Gateway.
inter.TransactionId = "12345678912345678912345678912345";
//End Point Url of the Gateway
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
```

Input-Incorrect Way of specifying values in Document Poll

The following lines of code show the incorrect property settings while invoking the DocumentPoll() on SAP generic connector. The comments provide the reason for the errors that arise due to incorrect property settings.

```
//InCorrect way of specifying the values
//Error as the ID is not 32 bit hexadecimal.
inter.TransactionId = "1234444";
//Will give an error if provided value as received for particular class does'nt match.
inter.CorrelationID = "12345678912345678912345678912335";
//Will give an error as value is blank,Should provide correct value for class as provided in asynchronous
submitRequest.
inter.ClassValue = "";
//Incorrect value of end Point Url of the Gateway
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
inter.documentPoll ( ref inter );
```

Output-Response for Document Poll

The following is the output of DocumentPoll() in case of incorrect property settings. The output status will generate an error with an error code and error list. The explanation for the error code can be looked up in the error codes list specified in Appendix A or in the

ErrorList given in the output.

```
//As the parameter is passing by ref ,the values for output can be retrieved from the same object which is passed
as parameter with all input values.
//Status of submission
//Output Values
//if submission fails at Gateway
```

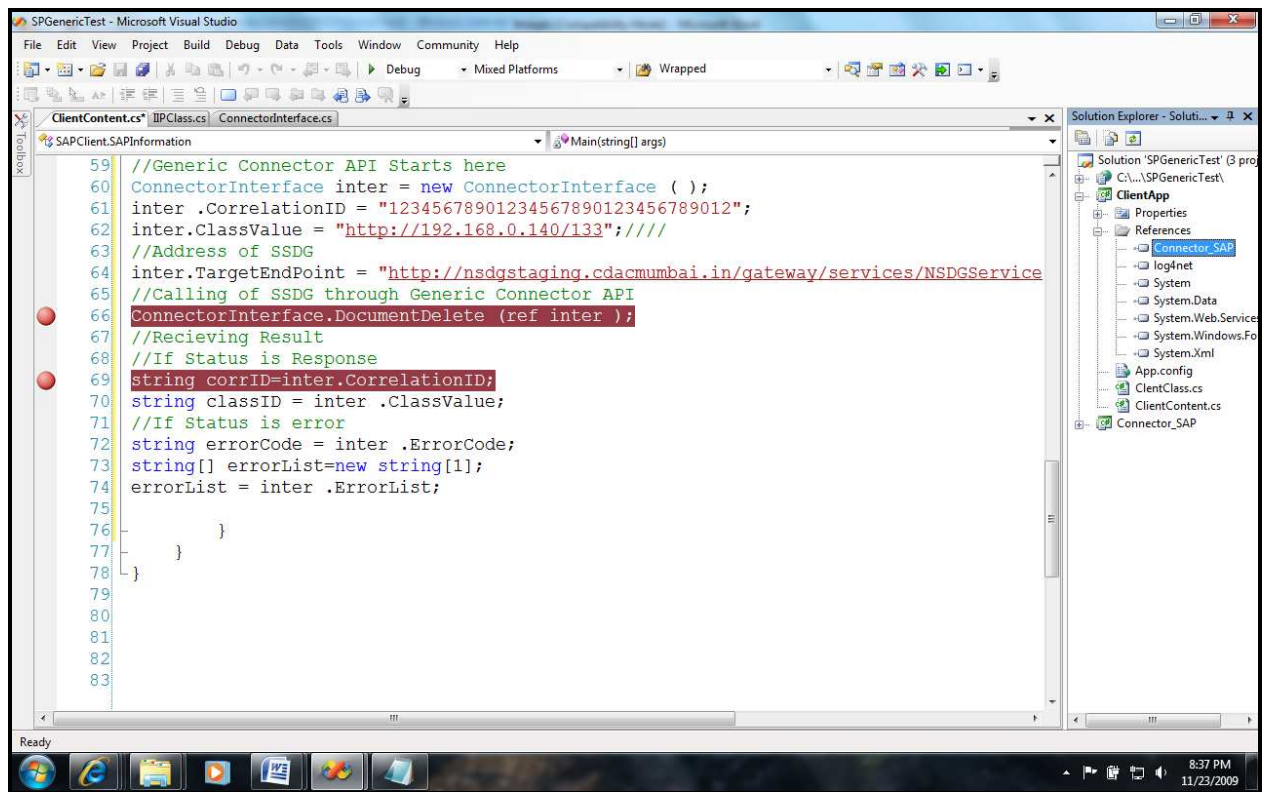
```
inter.SubmissionStatus = "error";  
//Error code detailed description related to error code can be viewed using the Table given in Appendix A or  
retriving the error description in the errorList field.  
inter.ErrorCode = "1033";  
//Reason for the error.  
inter.ErrorList = "Error specifying the reason";  
//If provided by the client.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";
```

iii) DocumentDelete

Input-Correct way of specifying values in Document Delete

The following lines of code show the correct property settings while invoking the DocumentDelete() on SAP generic connector. All properties pertaining to DocumentDelete() are explained in Table 10. The comments provide the description of the properties

```
//Set Values in the Object of API provided  
ConnectorInterface inter = new ConnectorInterface ();  
inter.TransactionId = "12345678912345678912345678912345";  
//or  
//Will give response as it is optional field  
inter.TransactionId = "";  
//It should contain the value as received in SubmitPoll  
inter.CorrelationID = "12345678912345678912345678912345";  
//End Point URL for the service which is used in asynchronous SubmitRequest  
inter.ClassValue = "http://127.0.0.1/22";  
//End Point Url of the Gateway where documentDelete can be done  
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";  
inter.documentDelete ( ref inter );  
//Apart from the above values specify correct entries in App.config file also.
```



Output-Response for Document Delete

The following is the output of DocumentDelete(). If the property settings are correct, the output status will be an acknowledgement for the DocumentDelete().

```
//Response for Correct document delete and succesful submission at Gateway.
//As the parameter is passing by ref ,the values for output can be retrived from the same object which is passed
  as parameter with all input values.
//Status of submission
//Output Values
//if the submission status is response,it implies that Gateway has succesfully deleted the request sents by SAP.
inter.SubmissionStatus = "response";
//Valid 32 Bit value provided by gateway. inter.CorrelationID = "12345678912345678912345678912345";
//If provided by the client.It remains Unchanged from Gateway.
inter.TransactionId = "12345678912345678912345678912345";
//End Point Url of the Gateway
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
```

Input-Incorrect Way of specifying values in Document Delete.

The following lines of code show the incorrect property settings while invoking the DocumentDelete() on SAP generic connector. The comments provide the reason for the errors that arise due to incorrect property settings.

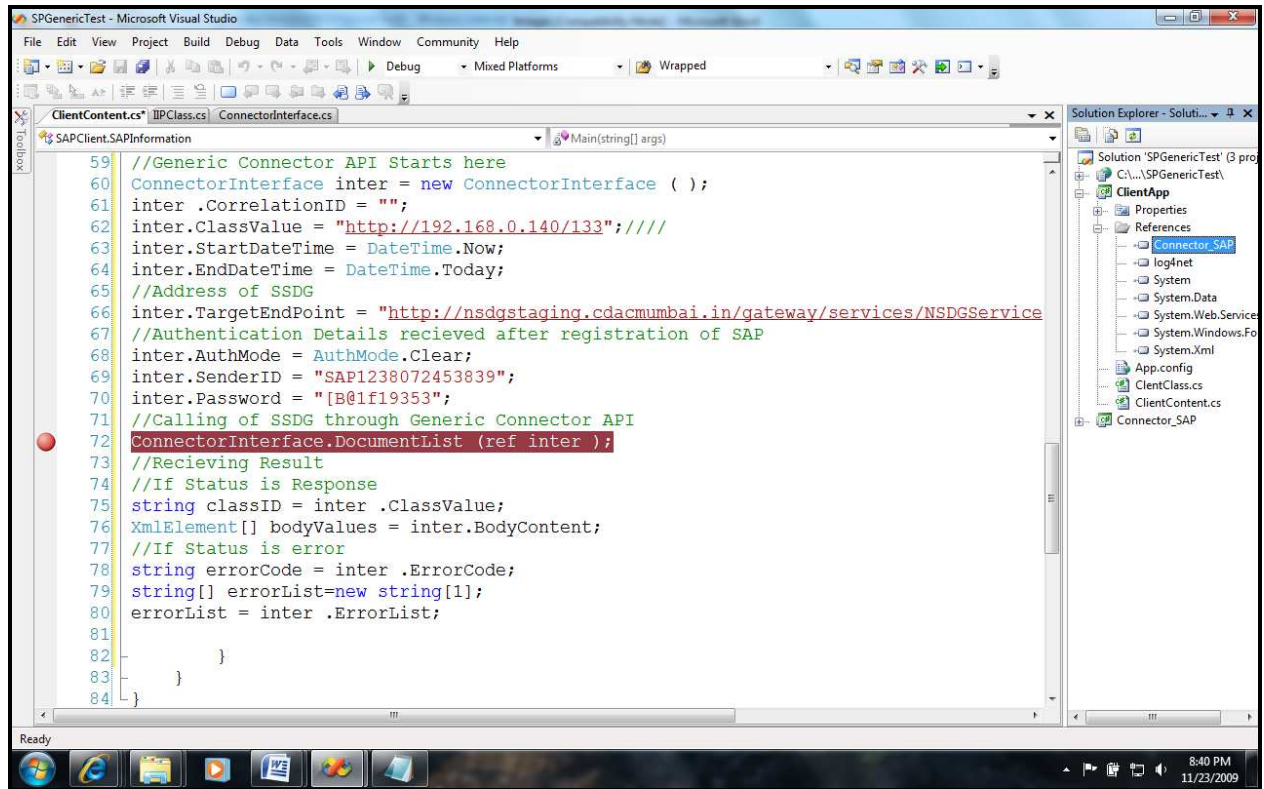
```
//Error as the ID is not 32 bit hexadecimal.  
inter.TransactionId = "1234444";  
//Will give an error if provided value as received for particular class does'nt match.  
inter.CorrelationID = "12345678912345678912345678912335";  
//Will give an error as value is blank,Should provide correct value for class as provided in submitRequest.  
inter.ClassValue = "";  
//Incorrect value of EndPoint URL  
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";  
inter.documentDelete ( ref inter );
```

Output-Response for the DocumentDelete

The following is the output of DocumentDelete() in case of incorrect property settings. The output status will generate an error with an error code and error list. The explanation for the error code can be looked up in the error codes list specified in Appendix A or in the ErrorList given in the output.

```
//Output Values  
//if submission fails at Gateway fails  
//or  
inter.SubmissionStatus = "error";  
//Error code detailed description related to error code can be viewed using the Table given in Appendix A or  
retriving the error description in the errorList field.  
inter.ErrorCode = "1033";  
//If Submission Status is error,reason for the error.  
inter.ErrorList = "Error specifying the reason";  
//If provided by the client.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";
```

iv) DocumentList



Input-Correct way of specifying values in Document List

The following lines of code show the correct property settings while invoking the DocumentList() on SAP generic connector. All the properties pertaining to DocumentList() are explained in Table 12,14 and 16 for Clear text, SHA1 and digital signature authentication modes respectively. The comments provide the description of the properties.

```
//Set Values in the Object of API provided
ConnectorInterface inter = new ConnectorInterface ( );
//Correct way of specifying the values
inter.TransactionId = "12345678912345678912345678912345";
//or
inter.TransactionId = ""; //Will give response as it is optional field
//It should be left blank for List Request
inter.CorrelationID = ""; //Correct way of specifying

//Response Mode of the service for which status is sought.
inter.ResponseMode = ResponseMode.Asynchronous; //Asynchronous Mode
//or
//Response Mode of the service for which status is sought.
inter.ResponseMode = ResponseMode.Synchronous;
//URL as per response mode for which status is sought
inter.ClassValue = "http://127.0.0.1/22";
```

```
//Depending upon the type of Authentication used by Service.
inter.AuthMode = AuthMode.Clear;
//or
inter.AuthMode = AuthMode.SHA1;
//or
inter.AuthMode = AuthMode.W3CSigned;
//If W3CSigned provide Keystore path in Configuration file for certificates used for signing the document.
//SenderID for Authentication Purpose
inter.SenderID = "SAP130720081454";
//Password for Authentication Purpose.Only applicable for clear and hash types of authentication.
inter.Password = "[B@13a87a0";
//End Point Url of the Gateway.
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
//XML Content in the form of XMLElement[].For List request Body content should be as per schema
mentioned in Appendix D
inter.BodyContent = body;
//Final Submission to SPGeneric Connector.ListRequest method is getting called.
inter.ListRequest ( ref inter );
//Apart from the above values specify correct entries in App.config file also.
```

Output-Response for List Request

The following is the output of DocumentList(). If the property settings are correct, the output status will be an acknowledgement for the DocumentList().

```
inter.SubmissionStatus = "response";
//If response take the response as per List response schema mentioned in Appendix D.
inter.BodyContent = body;
inter.CorrelationID = "12345678912345678912345678912345";
//If provided by the client.It remains Unchanged from Gateway.
inter.TransactionId = "12345678912345678912345678912345";
```

Input-Incorrect Way of specifying values in Document List

The following lines of code show the incorrect property settings while invoking the DocumentList() on SAP generic connector. The comments provide the reason for the errors that arise due to incorrect property settings

```
//Error as the ID is not 32 bit hexadecimal.
inter.TransactionId = "1234444";
//Will give an error as this should be left blank;
inter.CorrelationID = "12345678912345678912345678912345";
//Will give an error as value is blank,Should provide correct value for class.
inter.ClassValue = "";
//Will Give an error,should provide correct value for senderID.
inter.SenderID = "";
//Will Give an error,should provide correct value for password
inter.Password = "";
//Should provide correct end Point Url of the Gateway
```

```
inter.TargetEndPoint = "http://localhost/SPService;  
//XML Content in the form of XMLElement[] For List request Body content should be as per schema  
mentioned in Appendix D  
inter.BodyContent = body;  
//Final Submission to SPGeneric Connector.ListRequest method is getting called.  
inter.ListRequest ( ref inter );  
//Apart from the above values specify correct entries in App.config file also.  
//Current value in config files are example values and can be used.
```

Output-Response for DocumentList

The following is the output of DocumentList() in case of incorrect property settings. The output status will generate an error with an error code and error list. The explanation for the error code can be looked up in the error codes list specified in Appendix A or in the ErrorList given in the output

```
//Output Values  
//if submission fails at Gateway fails  
inter.SubmissionStatus = "error";  
//Error code detailed description related to error code can be viewed using the Table given in Appendix A or  
retriving the error  
description in the errorList field.  
inter.ErrorCode = "1033";  
//If Submission Status is error,reason for the error.  
inter.ErrorList = "Error specifying the reason";  
//If provided by the client.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";
```

5.2 SP Connector

i) asynSubmitRequest

Values received from SSDG can be retrieved in following parameters and class SPIInterface needs to be implemented as per application specific logic. Response Mode for this application is of Asynchronous type.

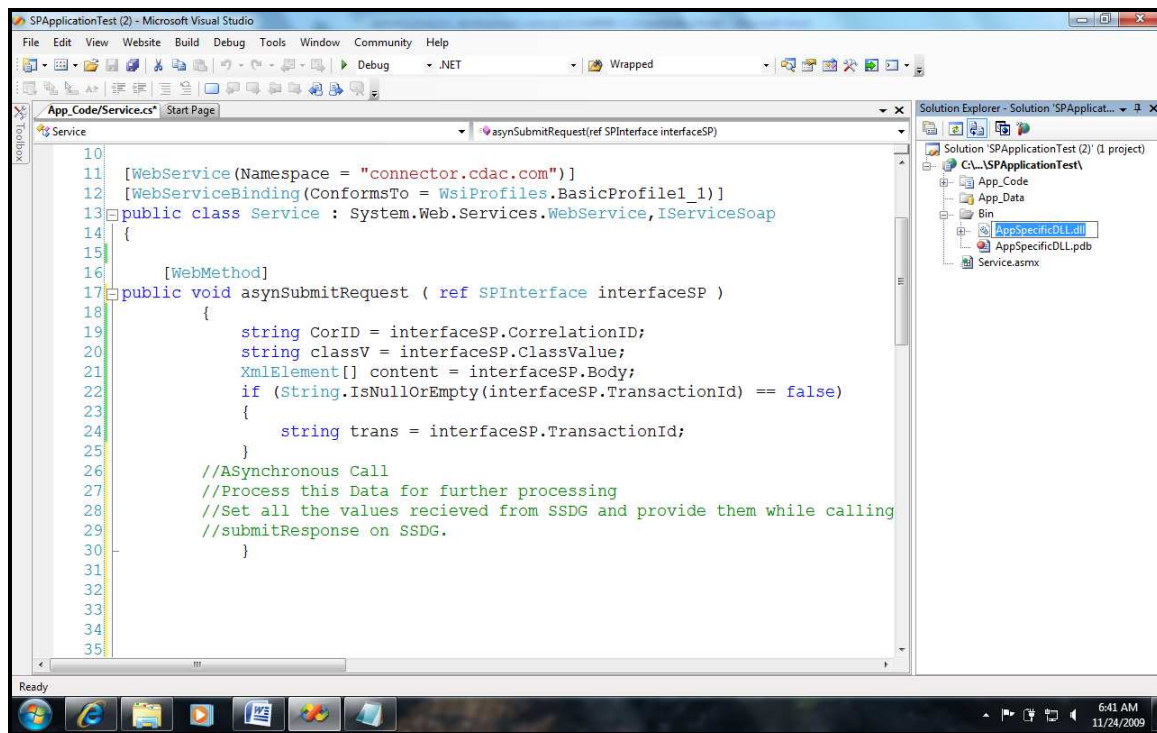
Input-Correct Way of Receiving the Input.

The following lines of code show the correct property settings are retrieved when asynSubmitRequest() is invoked on SP application connector by SP generic connector. All the properties pertaining to asynSubmitRequest() are explained in Table 27. The comments specify the properties that will be retrieved.

Method Call- asynSubmitRequest (ref SPInterface interfaceValues)

```
SPInterface inter=new SPInterface();  
//Retrieve TransactionID if set by SAP  
if ( String.IsNullOrEmpty (interfaceValues.TransactionID ) == false )  
inter.TransactionId = interfaceVaues.TransactionID;  
}  
//Retrieve Correlation ID  
inter.CorrelationID = interfaceValues.CorrelationID;  
// //Retrieve ClassValue  
inter.ClassValue = interfaceValues.ClassValue;  
//Retriev Body Content  
inter.BodyContent = interfaceValues.Body;  
//Retrieve Target End Point where the response needs to be submitted  
inter.TargetEndPoint = interfaceValues.TargetEndPoint;
```

Body Retrieved will be used by application for further processing,other information needs to be stored somewhere in order to put them at the time of submitResponse in asynchronous mode of requests.



ii) synSubmitRequest

Values received from SSDG can be retrieved in fields specified in SPInterface Class and method needs to be implemented as per application specific logic. Response Mode for this application is of Synchronous type.

Input-Correct Way of Receiving the Input.

The following lines of code show the correct property settings are retrieved when synSubmitRequest() is invoked on SP application connector by SP generic connector. All the properties pertaining to synSubmitRequest() are explained in Table 28. The comments specify the properties that will be retrieved.

Method Call- synSubmitReques (ref SPInterface interfaceValues)

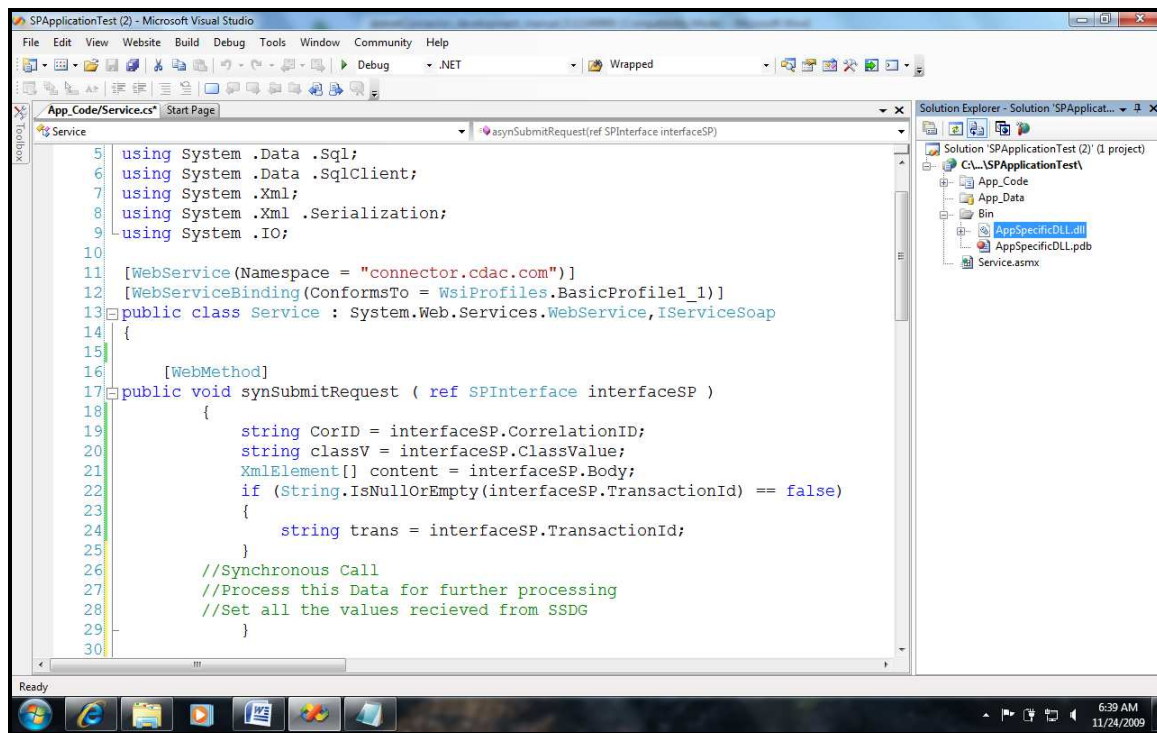
```
SPInterface inter=new SPInterface();  
//Retrieve TransactionID if set by SAP  
if ( String.IsNullOrEmpty (interfaceValues.TransactionID ) == false )  
  
    {  
        inter.TransactionId = interfaceVaues.TransactionID;  
    }  
//Retrieve Correlation ID  
inter.CorrelationID = interfaceValues.CorrelationID;
```

.Net Connector Development Manual

```
//Retrieve ClassValue
inter.ClassValue = interfaceValues.ClassValue;
//Retriev Body Content
inter.BodyContent = interfaceValues.Body;
//Retrieve Target End Point
inter.TargetEndPoint = interfaceValues.TargetEndPoint;
```

Body Retrieved will be used by application for further processing in synchronous manner. After processing the content specify values back as received from the gateway for synchronous mode. SP needs to mention the submission status in order to specify the response or error if business related in the document received from SAP.

```
//Set CorrelationID as set by gateway
inter.CorrelationID = "Specify value as received from gateway"; //Set ClassValue as set by gateway
inter.ClassValue = "Specify value as received from gateway";
//Set TargetEnd point as set by gateway or to which SP Service is registered
inter.TargetEndPoint = "Specify value as received from gateway";
//If body given by SAP gave correct information sought and correct
inter.SubmissionStatus = "response";
//If body given by SAP gave incorrect information sought and incorrect
inter.SubmissionStatus = "error";
//XML Content in the form of XElement[]. For business error SP should specify content as per schema
mentioned in Appendix D.
inter.BodyContent = body;
//Transaction ID provided by the SAP.
inter.TransactionId = "12345678912345678912345678912345";
```



iii) SubmitResponse

Results for Asynchronous mode of request can be submitted through this function call and setting fields as mentioned in Table 25. This function is invoked on Generic connector with the help of *SPImpl* class.

Input- Correct Way of Submitting the Output.

The following lines of code show the correct property settings while invoking the submitResponse() on the SP generic connector. All the properties pertaining to submitResponse() are explained in Table 25. The comments provide the description of the properties.

Method - SubmitResponse (ref SPInterface interfaceValues)

```
SPInterface inter=new SPInterface();  
inter.TransactionId ="12345678912345678912345678912345";  
//Value as received in Aynchronous SubmitRequest  
inter.CorrelationID ="12345678912345678912345678912345";  
//Value as received in Aynchronous SubmitRequest  
  
inter.ClassValue = http://123.34.5.56/20;  
//Specify the status.  
inter.SubmissionStatus="response";  
//or  
inter.SubmissionStatus="error";  
//Put content for response after application specific processing.
```

```
//for business related error specify content as per buisness error schema mentioned in Appendix D.  
inter.BodyContent = interfaceValues.Body;  
//Value as received in Aynchronous SubmitRequest  
inter.TargetEndPoint = interfaceValues.TargetEndPoint;
```

Output-Response for the Submitted Response

The following is the output of SubmitResponse(). If the property settings are correct, the output status will be an acknowledgement for the SubmitResponse().

```
//Response for Correct Asynchronous Submit Response and succesful submission at Gateway.  
//As the parameter is passing by ref ,the values for output can be retrived from the same object which is passed  
  as parameter with all input values.  
//Status of submission  
//Output Values  
// if the submission is in Asynchronous mode  
inter.SubmissionStatus = "Acknowledgement";  
//Valid 32 Bit value provided by gateway.Should be given this ID in documentPoll and documentDelete  
methods.  
inter.CorrelationID = "12345678912345678912345678912345";  
//If provided by the SAP.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";  
//End Point Url of the Gateway where documentPoll,documentDelete and listRequest can be done  
inter.TargetEndPoint = "http://localhost/SPService/Service.asmx";
```

Input-Incorrect Way of specifying the values

The following lines of code show the incorrect property settings while invoking the SubmitResponse() on SAP generic connector. The comments provide the reason for the errors that arise due to incorrect property settings.

```
//InCorrect way of specifying the values  
//Error as the ID is not 32 bit hexadecimal.  
inter.TransactionId = "1234444";  
//Should be same as received from gateway.  
inter.CorrelationID = "12345678912345678912345678912345";  
//Will give an error as value is blank,Should provide correct value for class as received from gateway.  
inter.ClassValue = "";
```

```
//Will Give an error,should provide correct value for body in the form of XMLElement[].  
inter.BodyContent = body;  
//End Point Url of the Gateway  
inter.TargetEndPoint = "http://localhost/SPService;
```

Output-Response for the SubmitResponse

The following is the output of SubmitResponse() in case of incorrect property settings. The output status will generate an error with an error code and error list. The explanation for the error code can be looked up in the error codes list specified in Appendix A or in the ErrorList given in the output.

```
//As the parameter is passing by ref ,the values for output can be retrived from the same object which is passed  
as parameter with all input values.  
//Status of submission  
//Output Values  
//if submission fails at Gateway  
inter.SubmissionStatus = "error";  
//Error code detailed description related to error code can be viewed using the Table given in Appendix A or  
retriving the error description in the errorList field.  
inter.ErrorCode = "1033";  
//Reason for the error.  
inter.ErrorList = "Error specifying the reason";  
//If provided by the client.It remains Unchanged from Gateway.  
inter.TransactionId = "12345678912345678912345678912345";
```

5.3 Error Logs

Error Logging is very critical to trouble shooting and as a norm errors should be logged for any future problem fixing. Developer would be specifying the path where the errors will be logged and in case of any disruption in working of the system, these error logs shall be made available to the concerned personnel fixing the problem.

Appendix

Appendix A: Error Codes

1001

The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document.The necessary element for submission are missing,please refer schema for element presence for valid submission.
The processing of your document submission failed. Please re submit.



1002	Authentication Failure. The credentials submitted with the document are invalid.
1003	The submitted document contained a SOAP Header. Envelope. Gateway doesn't process messages with SOAP headers.
1004	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing ResponseMode field.
1006	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Key field.
1007	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing NSDGErrors field.
1008	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing NSDGTTest field.
1009	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Transformation field.
1010	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Keys field".
1011	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Role field.
1012	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing EmailAddress field.
1013	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Organisation field.
1014	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Correlation ID field.
1015	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing IDAuthentication field.
1016	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing Body field.
1017	The submitted XML document failed to validate against the NSDGMMessageEnvelop schema for this class of document. Missing NSDGTimestamp Field"

1034	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Incorrectly populated NSDGTimestamp field for this kind of Submission.”
1035	The submitted document contains an entry for ResponseEndPoint, which is a reserved system field. This field should be left blank.
1036	The submitted document contains an invalid entry for TransactionID.
1037	The submitted document contains an invalid entry for AuditID.
1038	The submitted document should not contain NSDGErrors Field
1039	When submitting a document to Gateway the only valid values for the Qualifier field are request poll, response or error
1040	The submitted document contains an invalid entry for EnvelopeVersion.
1041	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Empty or Incorrectly populated Class Field.
1042	The submitted document contains an invalid entry for Function. The only valid values for the Function field are submit, delete or list.
1043	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Empty or Incorrectly populated Transformation Field.
1044	The submitted document contains an invalid entry for EmailAddress. If the field Qualifier contains the value “poll”, then the field EmailAddress must not be populated.
1045	The submitted document contains an invalid entry for Organisation. If the field Qualifier contains the value “poll”, then the field Organisation must not be populated.
1046	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Empty or Incorrectly populated Key Field.
1047	Unable to retrieve data for the supplied CorrelationID. Please ensure the CorrelationID is correct, and that you have the required authentication credentials to poll this CorrelationID.
1048	The submitted document contains an invalid entry for CorrelationID. If the field Function contains the value delete, then the field CorrelationID must be populated.
1049	Unable to retrieve data for the supplied CorrelationID. Please ensure the CorrelationID is correct, and that you have the required authentication credentials to delete this CorrelationID.
1050	The submitted document contains an invalid entry for Function. If the field Qualifier contains the value poll, then the only valid value for the Function field is submit.

1051	The submitted document contains an invalid entry for StartDateTime and EndDateTime. You cannot have a greater StartDateTime than EndDateTime.
1052	The submitted document contains an invalid entry for any one of the following fields StartDateTime, EndDateTime.
1053	The submitted document contains an invalid entry for Method, or an inconsistent Value entry. This field must contain the value 'MD5', 'clear' or 'W3Csigned'. If W3Csigned is specified the Value tag must be omitted and a Signature block must be present
1054	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Empty or Incorrectly populated Role Field.
1055	The submitted document contains an entry in the Body field. This field must be left blank for this transaction type.
1056	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Incorrectly populated NSDGTTest Field.
1057	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document.The Target Details Should not be present for this kind of submission
1058	The supplied user credentials failed validation for the requested service.
1059	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Empty or Incorrectly populated ResponseMode field.
1060	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Empty or Incorrectly X509Data field.
1061	When communicating with Gateway via HTML you MUST specify a valid ResponseEndPoint.
1062	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Empty or Incorrectly populated IDAuthentication field.
1063	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Empty or Incorrectly Populated AuthTimestamp field in Authentication Block.
1064	The submitted document contains an entry for SenderDetails, which is a reserved system field . This field should be left blank for this kind of submission.
1065	The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing or Incorrectly populated Signature field.
1066	The submitted document contains an entry for NSDGDDetails, which is a reserved system field . This field should be left blank for this kind of submission.
1067	The submitted XML document failed to validate against the

	NSDGMessageEnvelope schema for this class of document. Incorrectly populated Location field
1068	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Missing or Incorrectly Populated Text field
1069	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Incorrectly populated Error field
1070	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Incorrectly populated RaisedBy field
The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Incorrectly populated Type field	
1072	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Incorrectly populated Number field
1073	The submitted XML document failed to validate against the NSDGMessageEnvelope schema for this class of document. Incorrectly populated Authentication field
1100-1103	Synchronous behavior - Polling
1100	The document you have submitted has not been processed yet. The system is currently experiencing high volumes. To view this submission later use the URL provided.
1101	You have exceeded your allowable number of poll attempts. To view this submission later use the URL provided.
1102	The time allowed for the poll attempts has elapsed. To view this submission later use the URL provided.
1103	The record you have attempted to poll has been marked for deletion and therefore cannot be retrieved.
1500-1504	Messages that refer to the Test Service
1501	The total number of bytes in a document submitted to the Test Service in Gateway must not exceed 1 megabyte.
1504	The authentication details in the NSDGMessageEnvelop Header do not match the credentials

	provided in the authentication to the Test Service.
2000-2005	Messages that refer to NSDG Services
2000	Gateway could not locate a record for the supplied Correlation ID - the submission may have been deleted or the Correlation ID may be invalid; If you have not received a response you should resubmit the document
2001	The document was an invalid size
2002	The document was an invalid size - not enough data was supplied
2003	The service associated with the submitted document type is currently unavailable
2005	Gateway has not received an acknowledgement of your submission from the Department within the permitted timescale. Please either resubmit or contact the Department directly to determine if your submission has been accepted
3000-3001	Messages that refer to Departmental services
3000	System Failure. The processing of this document has failed. Please resubmit.
3001	The submission of this document has failed due to departmental specific business logic in the <i>Body</i> .

Error codes raised by Service Providers

Error Code	Description
------------	-------------



6000-6016	Major tests
6000	"The processing of your document submission failed. Please re-submit."
6001	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document."
6002	"Authentication Failure. The credentials submitted with the document are invalid."
6003	"The submitted document contained a SOAP <i>Header.Envelope</i> . The Back Office Service doesn't process messages with SOAP headers."
6004	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>Qualifier</i> field."
6005	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated Envelope field."
6006	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>Class</i> field."
6007	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>Function</i> field."
6008	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>NSDGTest</i> field."
6009	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>Transformation</i> field."
6010	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>SenderID</i> field."
6011	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>Role.Value</i> field."
6012	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated <i>EmailAddress</i> field."
6013	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly

	populated Organisation field."
6014	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated CorrelationID field."
6015	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated Method field."
6016	"The submitted XML document failed to validate against the NSDGMessageEnvelop schema for this class of document. Missing, Empty or incorrectly populated Body field."
7000-7001	Messages that refer to Departmental services
7000	"System Failure. The processing of this document has failed. Please resubmit."
7001	"The submission of this document has failed due to departmental specific business logic in the Body ."

Appendix B: App.Config File for SAP Generic Connector

```
<?xml version="1.0" encoding="utf-8" ?>  
<configuration>  
  <configSections>  
    <section name="log4net"  
type="log4net.Config.Log4NetConfigurationSectionHandler,Log4net"/>  
  </configSections>
```

```
<log4net>
  <root>
    <level value="DEBUG" />
    <appender-ref ref="LogFileAppender" />
  </root>
  <appender name="LogFileAppender" type=
"log4net.Appender.RollingFileAppender" >
    <param name="File" value="C:\LOG\log.txt" />
    <param name="AppendToFile" value="true" />
    <rollingStyle value="Size" />
    <maxSizeRollBackups value="10" />
    <maximumFileSize value="10MB" />
    <staticLogFileName value="true" />
    <layout type="log4net.Layout.PatternLayout">
      <param name="ConversionPattern" value="%-5p%d{yyyy-MM-dd hh:mm:ss} - %m%n" />
    </layout>
  </appender>
</log4net>

<appSettings>
  <add key="KeysType" value="SAPNumber" />
  <add key="KeysValue" value="123456" />
  <add key="Role" value="role"/>
  <add key="EnvelopeVersion" value="1.0"/>
  <add key="EMail" value="xyz@cdacmumbai.in"/>
  <add key="Organisation" value="CDAC Mumbai"/>
  <add key="Transformation" value="XML"/>
  <add key="CertSubject" value="E=mohsinsutar@gmail.com, CN=mohsin sutar, OU=Terms of use at
www.safescrypt.com/rpa (c) 02, OU=Class 1 Consumer Individual Subscriber, O= Safescrypt"/>
</appSettings>
</configuration>
```

Following parameters need to be set by Application Specific.

- Log file-Where the logs needs to be stored.
- Certificate Path-If using XML signature verification.

Appendix C: Web.Config File for SP Generic Connector

```
<?xml version="1.0"?>
<!--
  Note: As an alternative to hand editing this file you can use the
  web admin tool to configure settings for your application. Use
  the Website->Asp.Net Configuration option in Visual Studio.
  A full list of settings and comments can be found in
  machine.config.comments usually located in
  \Windows\Microsoft.Net\Framework\v2.x\Config
-->
<configuration>
```

```
<configSections>
  <section name="log4net"
type="log4net.Config.Log4NetConfigurationSectionHandler,Log4net"/>
</configSections>
<log4net>
  <root>
    <level value="DEBUG" />
    <appender-ref ref="LogFileAppender" />
  </root>
  <appender name="LogFileAppender" type=
"log4net.Appender.RollingFileAppender" >
    <param name="File" value="log.txt" />
    <param name="AppendToFile" value="true" />
    <rollingStyle value="Size" />
    <maxSizeRollBackups value="10" />
    <maximumFileSize value="10MB" />
    <staticLogFileName value="true" />
    <layout type="log4net.Layout.PatternLayout">
      <param name="ConversionPattern" value="%-5p%d{yyyy-MM-dd hh:mm:ss} - %m%n" />
    </layout>
  </appender>
</log4net>
<appSettings>
  <add key="KeysType" value="SAPNumber" />
  <add key="KeysValue" value="123456" />
  <add key="Role" value="role"/>
  <add key="EnvelopeVersion" value="1.0"/>
  <add key="EMail" value="xyz@cdacmumbai.in"/>
  <add key="Organisation" value="CDAC Mumbai"/>
  <add key="Transformation" value="XML"/>
  <add key="deploymentURI"
value="http://localhost:1571/SPApplicationTest/Service.aspx"></add>
  <add key="LogFolder" value="C:\\Logs" />

  <add key="FileName" value="C:\\Logs\\ExceptionLog.txt" />
  <add key="Type" value="Fatal"/>

  <add key="Location" value="SP"/>
  <add key="RaisedBy" value="SP"/>

  <add key="GWText" value="The processing of your document submission
failed.Please re-submit."/>
  <add key="GWNumber" value="6000"/>

  <add key="AuthText" value="AuthenticationFailure,Configure Certificate
Properly"/>
  <add key="AuthNumber" value="6002"/>

  <add key="AuthFail" value="Failed to authenticate with provided
certificate,AuthenticationFailure"/>
  <add key="AuthFail" value="6002"/>

  <add key="SchemaText" value="XSD for Validation Not found,configure path in
configuration File.Schema Validation Failed"/>
</appSettings>
</configuration>
```

```
<add key="SchemaNumber" value="6001"/>
```

```
<add key="SyntaxText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document."/>
```

```
<add key="SyntaxNumber" value="6001"/>
```

```
<add key="ExceptionText" value="Unable to Call SPApplication"/>
```

```
<add key="ExceptionNumber" value="7000"/>
```

```
<add key="EVText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated Envelope field"/>
```

```
<add key="EVNumber" value="6005"/>
```

```
<add key="CText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated Class field"/>
```

```
<add key="CNumber" value="6006"/>
```

```
<add key="QText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated Qualifier field"/>
```

```
<add key="QNumber" value="6004"/>
```

```
<add key="FText" value="The Submitted document failed to validate against
```

```
the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated function field"/>
```

```
<add key="FNumber" value="6007"/>
```

```
<add key="CorText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated Correlation field"/>
```

```
<add key="CorNumber" value="6014"/>
```

```
<add key="RText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated response mode field"/>
```

```
<add key="RNumber" value="6017"/>
```

```
<add key="BText" value="The Submitted document failed to validate against the NSDGMessageEnvelope schema for this class of document.Missing, Empty or incorrectly populated Body field"/>
```

```
<add key="BNumber" value="6016"/>
```

```
<add key="ConfigText" value="configuration File Error"/>
```

```
<add key="ConfigNumber" value="7001"/>

<add key="CertSubject" value="E=mohsinsutar@gmail.com, CN=mohsin sutar, OU=Terms of use at
www.safescrypt.com/rpa (c) 02, OU=Class 1 Consumer Individual Subscriber, O= Safescrypt"/>
</appSettings>

<connectionStrings/>
<system.web>
<!--
Set compilation debug="true" to insert debugging
symbols into the compiled page. Because this
affects performance, set this value to true only
during development.
-->
<trace enabled="true" localOnly="false" />
<compilation debug="true"/>
<!--
The <authentication> section enables configuration
of the security authentication mode used by
ASP.NET to identify an incoming user.
-->
<authentication mode="Windows"/>
<!--
The <customErrors> section enables configuration
of what to do if/when an unhandled error occurs

during the execution of a request. Specifically,
it enables developers to configure html error pages
to be displayed in place of a error stack trace.

<customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
<error statusCode="403" redirect="NoAccess.htm" />
<error statusCode="404" redirect="FileNotFound.htm" />
</customErrors>
-->
</system.web>
</configuration>
```

Following parameters need to be set by Application Specific.

- Deployment URI is the path of application specific connector which needs to set by Application Specific.
- Log file-Where the logs needs to be stored.
- Certificate Path-If using XML signature verification

Appendix D: Error Schema

Schema files for retrieving results by SAP and SP for Business related problems, List Response and List Request.

Business Error Response Schema

```
<?xml version="1.0" ?>
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:gt="http://
www.mit.gov.in/eGov/schema/NSDG/ "
xmlns:err="http://www.mit.gov.in/eGov/schema/NSDG/errorresponse"
targetNamespace="http://www.mit.gov.in/eGov/schema/NSDG/errorresponse"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
id="Error_Response"
version="1.0">
<xsd:annotation>
<xsd:documentation>This schema is used for errors returned by business systems. In these
circumstances, the header contains a single Error element of type "business" and the Body contains
further information.</xsd:documentation>
</xsd:annotation>
<xsd:element name="ErrorResponse">
<xsd:complexType>
```

```
<xsd:element ref="err:Application" minOccurs="0" maxOccurs="1" />
<xsd:element name="Error" maxOccurs="unbounded">
<xsd:complexType>
<xsd:element name="RaisedBy" type="xsd:string" />
<xsd:element name="Number" minOccurs="0" maxOccurs="1" type="xsd:integer" />
<xsd:element name="Type" type="xsd:string" />
<xsd:element name="Text" minOccurs="0" maxOccurs="unbounded" type="xsd:string" />
<xsd:element name="Location" minOccurs="0" maxOccurs="1" type="xsd:string" />
<xsd:element ref="err:Application" minOccurs="0" maxOccurs="1" />
</xsd:complexType>
</xsd:element>
</xsd:complexType>
</xsd:element>
```

```
<xsd:element name="Application">
<xsd:complexType>
<xsd:any namespace="##any" processContents="lax" />
<xsd:anyAttribute namespace="##any" />
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

List Response Schema

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns="http://www.mit.gov.in/eGov/schema/NSDG/BODY/ListResponse"
targetNamespace="http://www.mit.gov.in/eGov/schema/NSDG/BODY/ListResponse"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0" id="NSDGBodySchemaListResponse">
<xsd:annotation>
<xsd:documentation>This schema is used for the Body elements for the List Response message
document</xsd:documentation>
</xsd:annotation>
<xsd:element name="StatusReport">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="SenderID" minOccurs="0" maxOccurs="1" type="xsd:string" />
<xsd:element name="StartTimeStamp" minOccurs="0" maxOccurs="1" type="xsd:dateTime" />
<xsd:element name="EndTimeStamp" minOccurs="0" maxOccurs="1" type="xsd:dateTime" />
<xsd:element name="StatusRecord" minOccurs="0" maxOccurs="unbounded">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="TimeStamp" minOccurs="0" maxOccurs="1" type="xsd:dateTime" />
```

```
<xsd:element name="CorrelationID" minOccurs="0" maxOccurs="1">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:length value="32" />
<xsd:pattern value="[0-9A-F]{0,32}" />
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element name="Status" minOccurs="0" maxOccurs="1" type="xsd:string" />

</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

List Request Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.mit.gov.in/eGov/schema/NSDG/BODY/ListRequest"
targetNamespace="http://www.mit.gov.in/eGov/schema/NSDG/BODY/ListRequest"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="1.0" id="NSDGBodySchemaListRequest">

<xsd:annotation>
<xsd:documentation>
This schema is used for the Body elements for the List
Request message document
</xsd:documentation>
</xsd:annotation>
<xsd:element name="StatusQuery" >
<xsd:complexType>
<xsd:sequence>
<xsd:element name="StartTimeStamp" minOccurs="0" maxOccurs="1" type="xsd:dateTime" />
<xsd:element name="EndTimeStamp" minOccurs="0" maxOccurs="1" type="xsd:dateTime" />
</xsd:sequence>
</xsd:complexType>
</xsd:element>
```

</xsd:schema>

