Prepared (Subject resp)
Michael Zhang

No.		
Approved (Document resp) / Checked	Date / Rev	Reference
	2015-02-10 PA1	

ESIF - Identity API

1 Architecture

The Identity API allows partners to query the end user profile from their operator account, and ECE exposed it to partners through RESTful API.

OpenID Connect is employed to offer the APIs for partners who want to handle with operator's account.

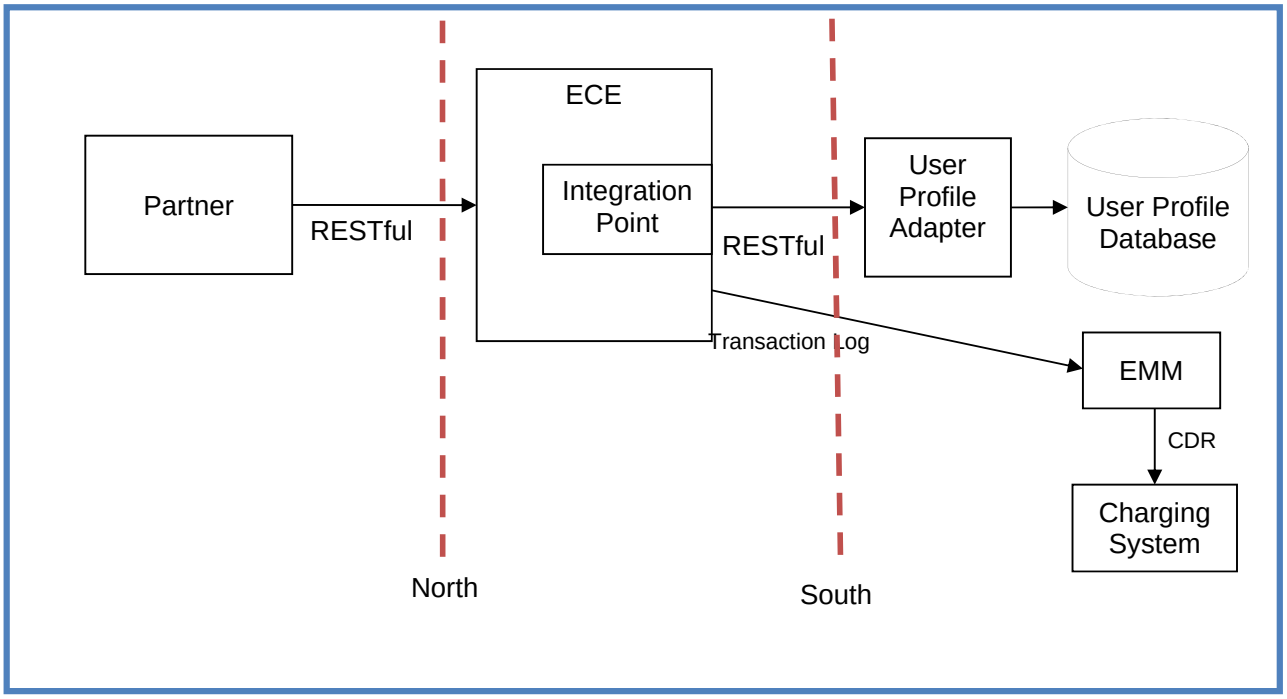

Figure 1 ESIF – Identity API Exposure Implementation

North Bound

On North Bound ECE expose the RESTful Identity API.

After purchased Identity API, the partner is able to:
- Authenticate the end user with Operator's account; and
- Get the profile information that the user registered in Operator.

South Bound

On South Bound, ECE support two interfaces.

1 ECE → User Profile Adaptor. ECE uses this HTTP/RESTful interface to perform the user profile query action corresponding to the North Bound Request. ECE relies on the

user profile adaptor which shall be developed by System Integrator to interoperate with specific customer's user profile database. Since the user profile databases are diverse in different customer, ECE does not provide any implementation to integrate user profile database, but the adaptor interface allows system integrator to integrate with specific customer database.

2 ECE → EMM. For offline changing and revenue settlement, ECE also need to generate transaction log for each API invoking transaction. The transaction logs are transfer to EMM where transaction logs are parsed and transformed to CDR. Eventually, CDRs are transfer to Charging System.

2 Security

ECE supports HTTP over SSL (HTTPS) to provide encryption and integrity protection the communication between Identity API Server and Client.

3 Authentication & Authorization

OAuth is used by ECE to authorize the access to identity API.

Before invoking the identity API, the client should obtained following data:

1 **Capability Access.**
Capability Access that consists of a couple of key and secret is used to authenticate the access to specific API. After partner purchased the API, partner is able to obtain the Capability Access for MSDP. Capability. Hereinafter, the term of AccessKey refers to the key of Capability Access.

2 **OAuth Registration**
Oauth registration is a registration of application describing an application that needs to access capability services authorized by an end user via OAuth, which includes a series of information relevant to OAuth. From OAuth Registration, partner is able to get

- Client ID: The client ID is generated after an application has been registered. The Client ID cannot be changed.
- Client Secret: The client secret is generated automatically after an application has been registered.

Client ID and Client Secret can be used to obtain OAuth Access Token as specific by RFC6749 [1] .

3.1 OAuth

The use case "Get User Profile" uses OAuth to obtain the authorization of end user (i.e. the resource owner in OAuth role) for the request asked by the partner.

In the process of OAuth, ECE plays as following roles:

- Resource Server: ECE exposes the Identity API and enforce AA for every access.
- Authorization Server: ECE opens Authorization Endpoint on which end users (i.e. resource owner) log in and grant authorization the client application. ECE also open

Token Endpoint on which client application exchanges the authorization code, client ID and client secret for an access token.

3.1.1 OAuth Authorization Portal

This is a user interface portal where the identity provider, i.e. the operator, can authenticate the user by username and password. In front-end, Auth Portal communicates with end-user's user agent, e.g. web browser. The OAuth Portal is responsible to display user interaction interface to end users and translates user's action, e.g. sign in, into requests to the Authorization Endpoint in back-end.
Note: **OAuth Portal is not a part of ECE product. Operator is responsible to implement it and integrate it to ECE.**

The ECE authorization server provides a built-in authorization portal as a reference implementation.

The authorization portal is responsible to:

- Authentication end user through a login page.
- Show the requested access scopes to the authenticated end user on authorization page. On this page, the end user needs to grant permission to the requesting application to access the API resource within allowed scope.

The interaction between APP used by end user and authorization portal is shown as following figure.

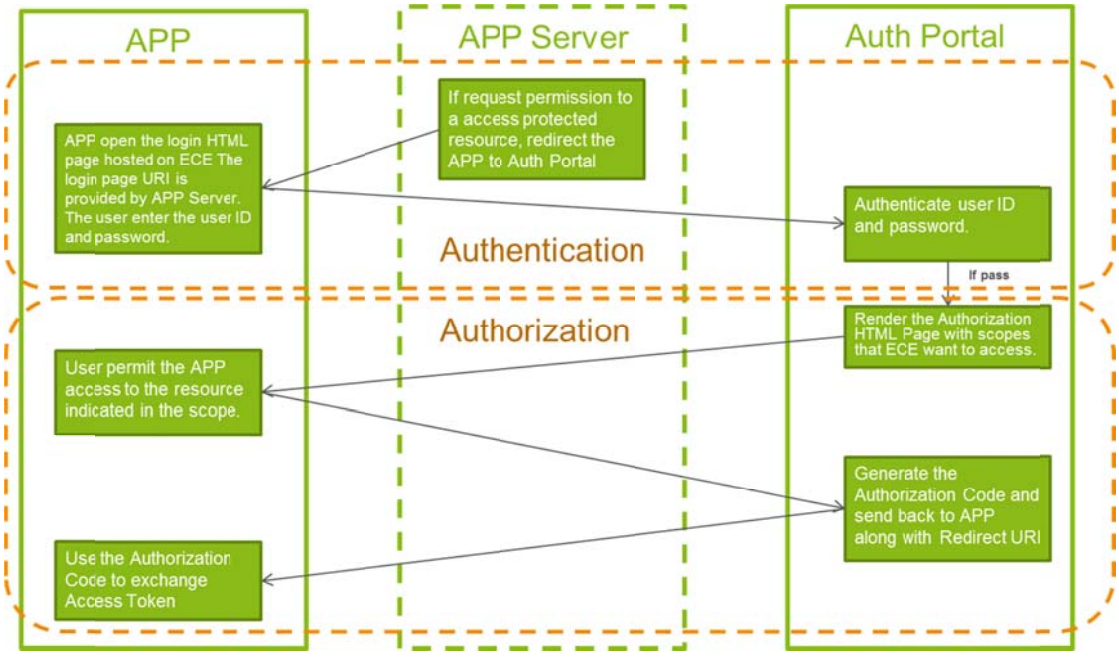

Figure 2 Interactions between APP and Authorization Portal

3.1.1.1 Access Authorization Portal

When the user chooses an operator as identity provider, the APP need to load the Authorization Portal to get the permission to use the identity API to execute the query transaction.

To load and render the authorization portal pages, the APP need to use web browser engine, e.g. WebKit, to access the authorization portal entry URL.

The entry URL shall conform to following ABNF definition

authPortalUri = “https:” “//” {server-root} “/oauth2/v1/authorize” “?” query_parms

query_parms = parameter *(“&” parameter)

paratmeter = clientID/
scopes/
responseType/
redirectUri

clientID = “client_id” “=” {client_ID}

scopes = “scope” “=” scope *[SP scope]
scope = {scope_item} [scope_parms]
scope_parms = scope_parm *[{scope_parm}]
{scope_parm} = “(” {name} “:” {value} “)”

responseType = “response_type” “=” response_type

response_type = “code”

redirectUri = “redirect_uri” = {redirURI}

In which:

{client_ID}:	This OAuth client ID that is allocated to the request application by MSDP.
{scope_item}:	This is the OAuth scope that authorization server pre-defined. In case of Identity API, the scope shall include **openeid and profile**
{scope_parm}	This is parameterized attributes related to specific scope. Take the example of Payment, the attributes could be the amount of payment and currency type. It is optional.
{redirURI}	This is the direct URI as defined in RFC 6749. It shall be the same as the URI registered in MSDP.

Example:

GET https://147.128.67.162:18081/oauth2/v1/authorize?client_id=playground@esif&scope=payment.chargerefund[(Amount:1234)(Currency:USD)(Charge_to:1234567890)]&response_type=code&redirect_uri=https://147.128.67.162:18081/oauth2/playground/callback/playground@esif

3.1.2 Authorization Flow

OAuth supports authorization using authorization code to get access token.

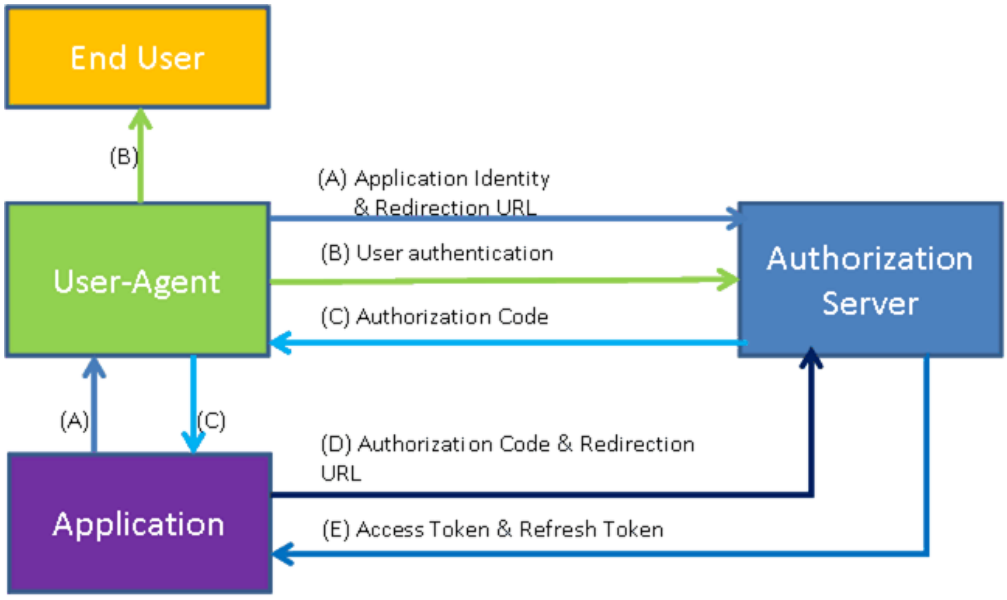

Figure 3 OAuth Authorization Flow

The authorization flow is as follows:

1 The application directs the user-agent of the end user to the authorization endpoint. The endpoint URL includes the application ID, the requested scope, the local state, and the redirection URI. The authorization server sends the user-agent back to the redirection URI once the access is granted or denied. See mark (A) in Figure 3.

2 The authorization server authenticates the end user (through the user-agent) using the authentication provider API. It sets up the authorization for the end user for granting application permissions to access resources. See mark (B) in Figure 3.

3 If the end user grants the access permission, the authorization server redirects the user-agent back to the application using the redirection URI. The redirection URI includes an authorization code and any local state provided by the application earlier. See mark (C) in Figure 3.

Note:

The redirection URI is provided in the request or earlier during application registration.

4 The application requests an access token from the authorization server token endpoint. The request includes the authorization code received in the previous step and the redirection URI used to obtain the authorization code for verification. See mark (D) in Figure 2.

5 The authorization server authenticates the application, validates the authorization code, and ensures that the received redirection URI matches the URI used to redirect the application in step3. If they are valid, the authorization server responds with an access token and a refresh token. See mark (E) in Figure 3.

Within the authorization flow, the client need to interwork with following OAuth Endpoint exposed on ECE

- Authorization Endpoint.
- Token Endpoint.

3.1.3 Authorization Endpoint

The client uses the authorization endpoint to interact with the resource owner and obtain an authorization grant.

For identity API, authorization code grant type is used. Therefore, client needs to invoke the **getAuthorizationCode** API to get authorization code.

Need details about Authorization Interface Specification

URI

Parameters

Example

3.1.3.1 Get Authorization Code API

The get authorization code operation is used to get authorization codes.

Resource and URI

Table 1 *Get Authorization Code Resource and URI*

URI	HTTP Method	Description

https://{serverRoot}/oauth2-api/i/v1/authorize	GET	Get authorization code.

Request

The following example shows a get authorization code request:

Example 1 *Get Authorization Code Request*

```
GET  https://ece.example.com /oauth2/v1/authorize?client_id=playground@partner&⇒
redirect_uri=https://www.client.com/callback&response_type=code&scope=location&⇒
state=playground@partner
```

The following table shows the parameters in the get authorization code request.

Table 2 Parameters in the Get Authorization Code Request

Parameters	Location	Description	Mandatory
response_type	Query	The value is code.	Yes
client_id	Query	It is made up of <*Service ID*>@<*Partner ID*> which can be obtained during registration. The maximum length is 101 characters.	Yes
redirect_uri	Query	It is one of the redirect_uri values during registration.	No
scope	Query	Space-separated list of scope keys defined in the system. Example value: ID_profile billing_information associated_lines	Yes
state	Query	Any string. It is used to prevent Cross-site scripting (XSS).	No

Response

The following example shows the response to a get authorization code request:

Example 2 Get Authorization Code Response

```
302 Found

Location: https://www.client.com/callback?state=playground@partner&code=6n2DZHn⇒
kZw6bWfp6OQqo
```

The following table shows the parameters in the response to a get authorization code request.

Table 3 Parameters in the Get Authorization Code Response

Parameters	Description
code	The authorization code.
state	If the state parameter is set, then the same state value is returned.

Error Response

The following example shows an error response to a get authorization code request:

Example 3 Get Authorization Code Error Response

```
400 Bad Request
Content-Type: application/json

{
  "error":"invalid_request",
  "error_description":"Required parameter 'scope' is missing.",
  "state":"playground@partner"
}
```

The following table shows the parameters in the error response to a get authorization code request.

Table 4 Parameters in the Get Authorization Code Error Response

Parameters	Description
error	Error code, the error codes list can be found in the Table 61.
error_description	It is an optional parameter. It is a human-readable UTF-8 encoded text, including, for example, the reason of the error.
state	It is required if state parameter exists in the client authorization request. It is the same as it in request.

The following table shows the possible error codes in the error response to a get authorization code request.

Table 5 Error Codes in the Get Authorization Code Error Response

Error Code	Description
invalid_request	Following cases are possible: • The request is missing a required parameter. • The request includes an invalid parameter value • The request is malformed.
access_denied	The resource owner or the authorization server denied the request.
unsupported_response_type	The authorization server does not support obtaining an authorization code using this response type.
invalid_scope	The requested scope is invalid, unknown, or malformed.
server_error	The authorization server encountered an unexpected condition which prevented it from fulfilling the request.

3.1.4 Token Endpoint

The client uses the token endpoint to obtain an access token by presenting its authorization code.

3.1.4.1 Exchange Access Token API

The exchange access token operation is used to exchange access tokens.

Resource and URI

Table 6 Exchange Access Token Resource and URI

URI	HTTP Method	Description
https://<serverRoot>/oauth2-api/p/v1/token	POST	Exchange the access token with code, or obtain a new access token using a refresh token.

Request

The following example shows an exchange access token request:

Example 4 Exchange Access Token Request

```
POST  /oauth2-api/p/v1/token HTTP/1.1
Host: ece.example.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic MkdPhwf2FCVm5rijcaCi

grant_type=authorization_code&code=6n2DZHnkZw6bWfp6OQqo&redirect_uri=https://ww⇒
w.client.com/callback
```

The following table shows the parameters in the exchange access token request.

Table 7 Parameters in the Exchange Access Token Request

Parameters	Location	Description	Mandatory
grant_type	Form	The value is authorization_code.	Yes
code	Form	The authorization code received from the authorization server. The length is 20 characters.	Yes
redirect_uri	Form	It is the same as it in the authorization request	Yes
Authorization	Header	It uses HTTP Basic to authenticate the client. The value must be: Basic <*identifier*> The <*identifier*> is a base64 encoded string literal, where the original is a clientid:password combination. The clientid is defined in Table 58. The password is specified by the developer during client registration.	Yes
client_id	Form	It is made up of <*Service ID*>@<*Partner ID*> which can be obtained during registration. The maximum length is 101 characters.	Yes, if HTTP Authorization header does not exist.
client_secret	Form	The client password during client registration.	Yes, if HTTP Authorization header does not exist.

Response

The following example shows the response to an exchange access token request:

Example 5 Exchange Access Token Request

```
200 OK
Content-Type: application/json
```

```
Cache-Control: no-store
Pragma: no-cache

{
  "access_token":"RhxVPlznUaTcYIxk2BG8",
  "token_type":"Bearer",
  "refresh_token":"7YdxmhSMyl3l1KOZw1e2",
  "scope":"location",
  "expires_in":3600
}
```

The following table shows the parameters in the response to an exchange access token request.

Table 8 Parameters in the Exchange Access Token Response

Parameters	Description
access_token	The access token. The length is 20 characters.
token_type	The value must be Bearer.
refresh_token	The refresh token. The length is 20 characters.
scope	It is the same as it in the request.
expires_in	The expired time of access token. The unit is second.

3.1.5 Use Case : GET User Profile

After getting OAuth Access Token, the client is able to invoke the Identity API exposed on ECE. The OAuth Access Token is populated into the HTTP Authorization header as specified in RFC6749.

Meanwhile, the ECE also need the access key to identify the subscription between partner and the API. The access key is put to the HTTP AccessKey header.

Following is the example of the request sent to ECE from Client.

Example 6 Request headers for OAuth

```
GET /rest/OpenIdConnect/userinfo
Authorization: Bearer MkdPhwf2FCVm5rijcaCi
AccessKey: ae23501
Host: ece.example.com:38080
Accept: application/json
```

4 Interface Specification

4.1 Northbound Identity API

4.1.1 General URI structure

Resource URI Pattern
resource_URI = api_root “/” resourcePath

In which

api_root = **schema** "//" **server "/" rootPath**

schema = **http-URI** /
https-URI

http-URI = "http:"

https-URI = "https:"

server = **host** [":" **port**]

rootPath = "/rest/OpenIdConnect/userinfo"

Example:

http://example.com/OpenIdConnect/userinfo

4.1.2 Content-Type

The Identity API supports application/JSON content types for GET operation. The response content type is application/JSON

4.1.3 Response Code

The following table shows some HTTP response codes and their indications.

Table 9 Response Codes

Response Codes	Indication	POST	GET	PUT	DELETE
200	The query is successful.		X		
400	Bad request. Check the error message and correct the request syntax.		X		
401	Authentication failure. Check the authentication requirements from your identity provider.		X		
403	Forbidden. Please provide authentication credentials.		X		
404	Not found: mistake in the host or path of the service URI.		X		
405	Method not supported. For example, only GET is supported in Identity API.		X		
422	Unprocessable Entity. For example, quota exceeded.		X		
500	Server busy and service unavailable. Please retry the request.		X		

4.1.4 Response Parameters

Table 10 Parameters for GetUserInfo Response

Parameter	Type	Description	Mandatory
sub	String	Subject - Identifier for the end user at the Issuer.	Yes

name	String	End user's full name in displayable form including all name parts, possibly including titles and suffixes, ordered according to the End-User's locale and preferences.	No
given_name	String	Given name(s) or first name(s) of the end user. Note that in some cultures, people can have multiple given names; all can be present, with the names being separated by space characters.	No
family_name	String	Surname(s) or last name(s) of the end user. Note that in some cultures, people can have multiple family names or no family name; all can be present, with the names being separated by space characters.	No
preferred_username	String	Shorthand name by which the end user wishes to be referred to at the RP, such as janedoe or j.doe. This value MAY be any valid JSON string including special characters such as @, /, or whitespace.	No
middle_name	String	Middle name(s) of the end user. Note that in some cultures, people can have multiple middle names; all can be present, with the names being separated by space characters. Also note that in some cultures, middle names are not used.	No
nickname	String	Casual name of the end user that may or may not be the same as the given_name. For instance, a nickname value of Mike might be returned alongside a given_name value of Michael.	No
profile	String	URL of the end user's profile page. The contents of this Web page SHOULD be about the End-User.	No
picture	String	URL of the end user's profile picture. This URL MUST refer to an image file (for example, a PNG, JPEG, or GIF image file), rather than to a Web page containing an image. Note that this URL SHOULD specifically reference a profile photo of the end user suitable for displaying when describing the end user, rather than an arbitrary photo taken by the end user.	No
website	String	URL of the end user's Web page or blog. This Web page SHOULD contain information published by the End-User or an organization that the end user is affiliated with.	No
email	String	End user's preferred e-mail address. Its value MUST conform to the RFC 5322 [RFC5322] addr-spec syntax.	No
email_verified	Boolean	True if the end user's e-mail address has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this e-mail address was controlled by the end user at the time the verification was performed. The means by which an e-mail address is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating.	No
gender	String	End user's gender. Values defined by this specification are female and male. Other values MAY be used when neither of the defined values are applicable.	No
birthdate	String	End user's birthday, represented as an ISO 8601:2004 [ISO86012004] YYYY-MM-DD format. The year MAY be 0000, indicating that it is omitted. To represent only the year, YYYY format is allowed. Note that depending on the underlying platform's date related function, providing just year can result in varying month and day, so the implementers need to take this factor into account to correctly process the dates.	No
zoneinfo	String	String from zoneinfo [zoneinfo] time zone database representing the End-User's time zone. For example, Europe/Paris or America/Los_Angeles.	No
locale	String	End user's locale, represented as a BCP47 [RFC5646] language tag. This is typically an ISO 639-1 Alpha-2 [ISO6391]	No

		language code in lowercase and an ISO 3166-1 Alpha-2 [ISO31661] country code in uppercase, separated by a dash. For example, en-US or fr-CA. As a compatibility note, some implementations have used an underscore as the separator rather than a dash, for example, en_US; Relying Parties MAY choose to accept this locale syntax as well.	
phone_number	String	End user's preferred telephone number. E.164 [E.164] is RECOMMENDED as the format of this Claim, for example, +1 (425) 555-1212 or +56 (2) 687 2400. If the phone number contains an extension, it is RECOMMENDED that the extension be represented using the RFC 3966 [RFC3966] extension syntax, for example, +1 (604) 555-1234;ext=5678.	No
phone_number_verified	Boolean	True if the end user's phone number has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this phone number was controlled by the End-User at the time the verification was performed. The means by which a phone number is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating. When true, the phone_number Claim MUST be in E.164 format and any extensions MUST be represented in RFC 3966 format.	No
address	JSON object	End user's preferred postal address. The value of the address member is a JSON [RFC4627] structure containing some or all of the members defined in Chapter 5.1.1 of OpenIDConnect core specification.	No
updated_at	Number	The time when the end user's information was last updated. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.	No

The following example shows the response to a query user request:

```
HTTP/1.1 200 OK
Date: Tue, 04 Sep 2014 08:05:03 GMT
Content-Type: application/json

{
 "sub":"usera”,
 "name":"Jane Ne Joe",
 "given_name":"Jane",
 "family_name":"Joe",
 "preferred_username":"j.doe",
 "middle_name":"Ne",
 "nickname":"Jane.d",
 "profile":"http://example.com/janedoe/profile",
 "picture":"http://example.com/janedoe/me.jpg",
 "website":"http://example.com",
 "gender":"Male",
 "birthdate":"1999-01-01",
 "locale":"en-US",
 "updated_at":1300000000,
 "email":"janedoe@example.com",
 "email_verified":true,
 "address":{
 "country":"China",
 "postal_code":"510000",
 "street_address":"X Street"
 },
 "phone_number":"+1 (425) 555-1212",
 "phone_number_verified":true
}
```

4.1.5 Exceptions

If the request is processed successfully, ECE will return the 200 OK HTTP status code.

If the request cannot be processed, ECE will return the other HTTP status codes, e.g. 400 Bad Request, 500 Internal Server Error, etc.

Following is an example of exception.

Example 7 Invalid Request

```
400 Bad Request
Date:  Tue, 17 Jul 2012 09:33:49 GMT
Content-Type:  application/json

{
"message":"Not contain 'openid' scope."
}
```

Following table shows the meaning of the parameters in the exception example.

Table 11 Exception Parameters

HTTP Status Code	errorCode	Description	HTTP Response Body
400	N/A	The syntax of the request is not correct.	{"message":"<Error Message>"}
422	6	Service capability throttling threshold is exceeded.	{"errorCode":"<Error Code>", "message":"<Error Message>"}
	26	Partner or application throttling threshold is exceeded.	
	32	Quota exceeded in either Application Subscription or Partner Subscription.	
	35	System parameter does not exist.	
500	1	Unexpected network or system error.	

4.2 Southbound User Profile Intetration Point

User Profile Integration Point is defined as a Hypertext Transfer Protocol (HTTP) base interface. ECE acts as an HTTP client and the integration adapter acts as an HTTP server.

The network channel between ECE and the integration adapter must be protected from the Internet.

The integration adapter can be deployed inside or outside ECE with multiple instances. All instances of the integration adapter must be reachable through the same URL.

4.2.1 Request

4.2.1.1 URI

Resource URI	HTTP Method	Operation
{queryUrl}?ownerId={ownerId}	GET	Get user profile

queryUrl shall be configured by system configuration parameter facility.userprofile.server.url from WebConsole.

4.2.1.2 Accept Content-Type

The user profile integration point supports application/JSON content types for GET reponse.

4.2.1.3 Request Example

Following is an example of the ECE request:

Example 8 query user profile request

```
GET /rest/queryuser?ownerId=usera
Host: ece.example.com:38080
Accept: application/json
```

4.2.2 Response Parameters

Table 12 Parameters for user profile adapter response

Parameter	Type	Description	Mandatory
sub	String	Subject - Identifier for the end user at the Issuer.	Yes
name	String	End user's full name in displayable form including all name parts, possibly including titles and suffixes, ordered according to the End-User's locale and preferences.	No
given_name	String	Given name(s) or first name(s) of the end user. Note that in some cultures, people can have multiple given names; all can be present, with the names being separated by space characters.	No
family_name	String	Surname(s) or last name(s) of the end user. Note that in some cultures, people can have multiple family names or no family name; all can be present, with the names being separated by space characters.	No
preferred_username	String	Shorthand name by which the end user wishes to be referred to at the RP, such as janedoe or j.doe. This value MAY be any valid JSON string including special characters such as @, /, or whitespace.	No
middle_name	String	Middle name(s) of the end user. Note that in some cultures, people can have multiple middle names; all can be present, with the names being separated by space characters. Also note that in some cultures, middle names are not used.	No
nickname	String	Casual name of the end user that may or may not be the same as the given_name. For instance, a nickname value of Mike might be returned alongside a given_name value of Michael.	No
profile	String	URL of the end user's profile page. The contents of this Web page SHOULD be about the End-User.	No
picture	String	URL of the end user's profile picture. This URL MUST refer to an image file (for example, a PNG, JPEG, or GIF image file), rather than to a Web page containing an image. Note that this URL SHOULD specifically reference a profile photo of the end user suitable for displaying when describing the end user, rather than an arbitrary photo taken by the end user.	No
website	String	URL of the end user's Web page or blog. This Web page SHOULD contain information published by the End-User or an	No

		organization that the end user is affiliated with.	
email	String	End user's preferred e-mail address. Its value MUST conform to the RFC 5322 [RFC5322] addr-spec syntax.	No
email_verified	Boolean	True if the end user's e-mail address has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this e-mail address was controlled by the end user at the time the verification was performed. The means by which an e-mail address is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating.	No
gender	String	End user's gender. Values defined by this specification are female and male. Other values MAY be used when neither of the defined values are applicable.	No
birthdate	String	End user's birthday, represented as an ISO 8601:2004 [ISO86012004] YYYY-MM-DD format. The year MAY be 0000, indicating that it is omitted. To represent only the year, YYYY format is allowed. Note that depending on the underlying platform's date related function, providing just year can result in varying month and day, so the implementers need to take this factor into account to correctly process the dates.	No
zoneinfo	String	String from zoneinfo [zoneinfo] time zone database representing the End-User's time zone. For example, Europe/Paris or America/Los_Angeles.	No
locale	String	End user's locale, represented as a BCP47 [RFC5646] language tag. This is typically an ISO 639-1 Alpha-2 [ISO6391] language code in lowercase and an ISO 3166-1 Alpha-2 [ISO31661] country code in uppercase, separated by a dash. For example, en-US or fr-CA. As a compatibility note, some implementations have used an underscore as the separator rather than a dash, for example, en_US; Relying Parties MAY choose to accept this locale syntax as well.	No
phone_number	String	End user's preferred telephone number. E.164 [E.164] is RECOMMENDED as the format of this Claim, for example, +1 (425) 555-1212 or +56 (2) 687 2400. If the phone number contains an extension, it is RECOMMENDED that the extension be represented using the RFC 3966 [RFC3966] extension syntax, for example, +1 (604) 555-1234;ext=5678.	No
phone_number_verified	Boolean	True if the end user's phone number has been verified; otherwise false. When this Claim Value is true, this means that the OP took affirmative steps to ensure that this phone number was controlled by the End-User at the time the verification was performed. The means by which a phone number is verified is context-specific, and dependent upon the trust framework or contractual agreements within which the parties are operating. When true, the phone_number Claim MUST be in E.164 format and any extensions MUST be represented in RFC 3966 format.	No
address	JSON object	End user's preferred postal address. The value of the address member is a JSON [RFC4627] structure containing some or all of the members defined in Chapter 5.1.1 of OpenIDConnect core specification.	No
updated_at	Number	The time when the end user's information was last updated. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.	No

The following is an example of user profile adaptor response:

```
HTTP/1.1 200 OK
```

```
Date: Tue, 04 Sep 2014 08:05:03 GMT
Content-Type: application/json

{
 "sub":"usera”,
 "name":"Jane Ne Joe",
 "given_name":"Jane",
 "family_name":"Joe",
 "preferred_username":"j.doe",
 "middle_name":"Ne",
 "nickname":"Jane.d",
 "profile":"http://example.com/janedoe/profile",
 "picture":"http://example.com/janedoe/me.jpg",
 "website":"http://example.com",
 "gender":"Male",
 "birthdate":"1999-01-01",
 "locale":"en-US",
 "updated_at":1300000000,
 "email":"janedoe@example.com",
 "email_verified":true,
 "address":{
 "country":"China",
 "postal_code":"510000",
 "street_address":"X Street"
 },
 "phone_number":"+1 (425) 555-1212",
 "phone_number_verified":true
}
```

4.2.3 Exceptions

If the request is processed successfully, the integration adapter must return the 200 OK HTTP status code.

If the request cannot be processed, the integration adapter can return the other HTTP status codes, e.g. 400 Bad Request, 500 Internal Server Error, etc. ECE handles all these status codes as 500 Internal Server Error and discards the body of the response.

5 Usage Data Record

ECE generates and writes transaction log for every access to the Identity API. These logs are written to ECE's file system. Later on, a mediation server, e.g. EMM, pull the transaction log file from ECE. Mediation server transform the transaction log into CDRs, and transfer CDRs to charging system where offline charging and billing are performed.

5.1 Transaction Log Format

The transaction log consists of the following three parts:

- Common header
- Functional body
- Customized body

The header part can be used for general traffic statistics, while the body parts can be used for business tracing.

All transaction information is written in the plain text log file. For a transaction log entry, all parts are written in the following format:

<Common Header>,<Functional Body>,<Customized Body>

For more details about format of ECE's transaction log please refer to [4].

5.2 Parameters

Existing ECE transaction log includes following parameters for every Identity API transaction. These parameters are contained in functional body of Identity transaction log.

5.2.1 ESIF Common Parameter

Following table lists the general parameters that are available in all ESIF related transaction logs.

Table 13 Transaction Log Customized Body

Parameter	Description	Source of data	Location
externalRatingServiceKey	The cross reference of the actual product offering instance being maintained in Service Registry of order Care. It can be mapped to the ADMIN_External_RatingService_Key attribute in charging system offer.	Provisioned from MSDP	Customized Body
apiIdentifier	The service capability ID of the API on-boarded on ECE.	ECE provisioning	Customized Body
partnerMsisdn	The dummy MSISDN assigned to a specific partner. It is provided from MSDP.	Provisioned from MSDP	Customized Body
PartnerID	Partner Identity which is provisioned from MSDP	Provisioned from MSDP	Common Header: ServiceProviderId
AccessKey	Capability Access Key used in the API Access Request	Received Request	Common Header: ApplicationId

5.2.2 Identity API Specific Parameter

Following table lists the parameters specific for Identity API's transaction information.

Table 14 *Transaction Log Functional Body*

Parameter	Description	Source of data	Location
token	The authenticate token used for authentication.	Request authorization header	Functional Body
scopes	Space-separated list of the scopes in the token	Oauth response	Functional Body
ownerId	The owner id of the queried user profile	Oauth response	Functional Body

5.3 Log Format

The ESIF specific parameters will be put to the Customized Body part in the transaction log.

ESIF need to implement Integration Point adaptor to customize the existing transaction log.

5.4 Log Files

The transaction logs relevant to identity API are written the following log files in the directory <*SIG_ROOT*>/esif/statisticslog/identity. For more details about ECE transaction log files, please refer to [4] .

- GetUserInfo.log.[time]
  This file contains the transaction logs generated by GetUserInfo operation.

6 Entity Definition

For ESIF adaption purpose, some ESIF specific properties shall be added to ECE entity model, as shown in following figure.

6.1 Service Partner

After installed ESIF pre-integrated API package onto ECE, an additional customized property is added under the **ServiceProvider** entity.

Property Identity	Display Name	Type	category	Description
Extension	Partner MSISDN	Non Empty String	Mandatory	The dummy MSISDN assigned to the ESIF partner.

6.2 Service Capability

Service Capability Name: **OpenIdConnect**

The ECE built-in service capability **OpenIdConnect** is exposed as the ESIF pre-integrated identity API.

6.3 Service Provider Service Subscription(SPSS)

After installed ESIF pre-integrated API package onto ECE, two additional customized properties as below are added under the OpendIdConnect **Service Provider Service Subscription** entity that represent the purchasing relationship between partner and identity API..

Property Identity	Display Name	Value Type	Property Type	Description
ExternalRatingServiceKey	External Rating Service Key	Non Empty String	Mandatory	Specifies the unique service key assigned to each partner's product (e.g. Service key for "SMS Bronze" API product instance). It is used to select the correct product instance to be used for that product specific service consumption during rating and reported in the Usage report/CDR/Rating Trigger sent to Charging System.
APIType	API Type	Non Empty String	Mandatory	Specifies the API type. The type is decided based on how this API is used(as management or real-time API) according to the value set in EOC

6.4 Application Service Subscription (APPSS)

OpenIdConnect **ApplicationServiceSubscription** entity is not customized for ESIF.

7 Integration

7.1 OAuth Authorization Server Integration

The ECE OAuth authorization server provides authentication and authorization for ECE Identity Service and any external client. It is based on the OAuth 2.0 authorization framework.

The OAuth authorization server is integrated with the user authentication provider to handle user authentication. Authenticating the end user in the OAuth authorization portal can be customized for adapting to different authentication methods. The OAuth authorization server can authorize the client against the end user, supporting user granted client permission for accessing the resources. For more information about OAuth authorization server is integration, refer to Ericsson Composition Engine, Authorization Integration Guide[11].

The OAuth authorization server supports the Integration Point 3.0 integration framework for system integration and customization. For more information about Integration Point 3.0, see Ericsson Composition Engine, Service Exposure Integration Point Development[10].

7.2 User Profile Adapter Integration

In this integration, the user profile adapter URL is configured in system configuration parameter facility.userprofile.server.url through ECE Web Console. Following figure shows an example of the configuration.

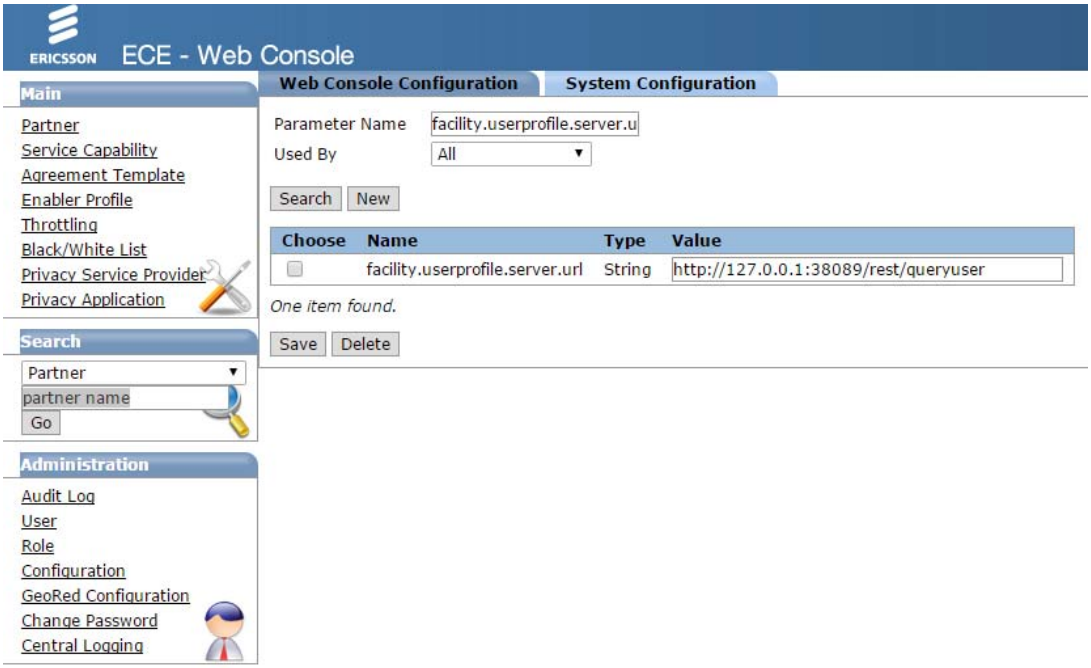

Figure 4 Configure User Profile Adapter URL through Web Console.

8 Installation Preparation

8.1 System Planning

8.1.1 ESIF DP Group and DPs Mapping

DP (deployment profile) and DP group are new concepts introduced in ECE 15.Refer to *Ericsson Composition Engine, Installation Instruction [7]* for details.

ESIF DP group has several DPs. The mapping is as below:

DP Group	DPs	Installation Playlist
ESIF	DP-ESIF-Traffic	Maiden_Install_ESIF.xml
	DP- ESIF-LBS	

8.1.2 ESIF DPs Configuration

DP Name	LBS Host Name	HTTP Port	Admin Port
DP-ESIF-Traffic	-	27090	27092
DP- ESIF-LBS	vip-dp-esif-lbs	-	-

8.1.3 Rules of ESIF DPs Deployment

ESIF follows the general DP deployment rules defined in *Ericsson Composition Engine, Installation Instruction [7].*

8.2 Creating Backup

It is suggested the system integrator create backup before the ECE installation.

9 Installation

9.1 Installation Prerequisites

Before starting the installation, make sure the following prerequisites are met.

Note: The operator must get the user authority to the occas installation directory in the operating system.

- The basic environment of Ericsson Composition Engine Generic Package

10 (GP10) is set up. For details, refer to Ericsson Composition Engine, Installation Instruction.

- The system is healthy, both on hardware and software levels.
- Following DP groups are installed: AC, Exposure, Foundation and OAuth.

9.2 ESIF Installation

To install ESIF, do the following:

1. Prepare the following installation packages:
    - ESIF CXP Package: 19089_CXP9040466_X-A.tar.gz
    - ESIF DP RPMs:

- DP-ESIF-LBS-1.0.0.rpm
- DP-ESIF-Traffic-1.0.0.rpm

- ECE Maiden Playlist: SUF_Repository_ECE-3.0.1.tar.gz, which contains ESIF Maiden playlist: Maiden_Install_ESIF.xml.

2. Import ECE Maiden playlist to the SUF repository.
3. Import all the RPMs to YUM Repository:
   - Upload ESIF CXP 19089_CXP9040466_X-A.tar.gz to the YUM repository.
   - Extract the CXPs with tar command.
   - Copy all the RPMs in CXP to the YUM repository directory.
   - Upload ESIF DP RPMS to the YUM repository directory.
   - Update the YUM repository Meta-data.
   - Update /etc/yum.repos.d/rda-repo.repoin all blades to include above YUM repository.
4. Update the Network Config File to configure ESIF DPs on ECE nodes from SUF.
5. Create a job with the updated NetWork Config File and Maiden_Install_ESIF.xml playlist from SUF.
6. Run the job to install ESIF.

10 Configuration Properties

10.1 Configure OpenIdConnect Service Capability

Do the following to configure OpenIdConnect Service Capability for ESIF:

1. Open the Web Console, from the Main field of Navigator, click Service Capability to enter the Service Capability List Tab page.
2. In the Already Imported field of Main Page, click OpenIdConnect service capability to enter the Service Capability Tab page.
3. In Main Page, check the Status check-box to set the service capability active.
4. In the Status drop-down list of Main Page, select Active to activate the service capability.
5. In the property fields of Main Page, check the check-boxes to perform the further configuration for the properties.

Note: The default options of these check-boxes are unchecked.

The following four mandatory properties must be set:

- **Default Currency**
- **Allow ChargeAmount**

The default value is False. Must be True for ESIF.

- **Allow RefundAmount**

The default value is False. Must be True for ESIF.

- **Allow ReserveAmountCharging**

The default value is False.

6. In the checked property fields of Main Page, set the value for each property.
7. In Main Page, click Save to activate all settings for the service capability.

10.2 Configure User Profile Adapter

Do the following to configure User Profile Adapter for ESIF OneApi Identity Service:

1. Open the Web Console, from the **Main** field of Navigator, click **Configuration** to enter the configuration page.
2. Chose **System Configuration** tab from the Main Page bottom, search and set Parameter "facility.userprofile.server.url".

11 REFERENCE

[1] RFC6749, The OAuth 2.0 Authorization Framework, https://tools.ietf.org/html/rfc6749;

[2] Authorization Integration Guide, 2/1553-CXP 904 0266 Uen;

[3] RFC2617, HTTP Authentication: Basic and Digest Access Authentication, https://tools.ietf.org/html/rfc2617;

[4] Service Exposure Transaction Log Description, 16/198 17-CXP 904 0189 Uen;

[5] Ericsson Composition Engine, Installation Instruction, 1/155 42-APR 901 0383/G Uen.

[6] Ericsson Composition Engine - OpenID Connect Development Guide, 72/1553-CXP 904 0189 Uen