Linee Guida Attribute Authority – Allegato tecnico OAS3

Specifiche tecniche per la realizzazione dell'architettura Attribute Authority in SPID

Sommario

History of Changes

DATE	AUTORE	VERSIONE	MODIFICHE
31/05/2021	AGID	1.0	PRIMA VERSIONE

1. Architettura delle Attribute Authority

I soggetti che intendono operare come AA devono predisporre l'accesso ad una o più interfacce API RESTful[1] . L'interfaccia API deve essere implementata in accordo con le specifiche riportate nelle presenti linee guida e in conformità con il MODI[2]. I soggetti che intendono operare come AA e che necessitano l'acquisizione del consenso da parte dell'utente, oltre all'interfaccia API, devono esporre anche un livello software analogo ad un SP che permetta all'utente di effettuare l'autenticazione presso l'IdP al fine di ottenere il consenso da parte dell'utente a trasmettere gli attributi qualificati al SP richiedente.

2. Interfaccia API RESTful

L'interfaccia API che permette la comunicazione tra SP e AA deve essere implementata rispettando i criteri di progettazione definiti dal paradigma REST[1]. In particolare, secondo quanto previsto dal paradigma, le operazioni consentite devono essere *stateless*, devono essere mappate su *metodi HTTP* in funzione della loro tipologia e devono essere *orientate alla risorsa*. Il dialogo DEVE utilizzare esclusivamente il protocollo HTTPS in conformità con le Raccomandazioni AGID - TLS e Cipher Suite[3]. Per maggiori indicazioni in merito alle caratteristiche dello stile architetturale REST, si rimanda al MODI[2].

2.1 Estensioni OpenAPI

Il *documento OpenAPI* DEVE contenere l'elemento *"#/info/x-spid"* con i seguenti campi:

Campo	Tipo	Descrizione
aa-version	string	OBBLIGATORIO. Deve contenere il valore **1.0.0** quale riferimento alla versione delle linee guida SPID per le AA cui l'API fa riferimento. Tale valore non è da confondere con il valore nel campo openapi che specifica la versione OpenAPI, né con il valore nel campo version dell'oggetto info, che invece specifica la versione del documento.
aa-home	string (url)	OBBLIGATORIO. URI presso l'AA dove è esposto il presente *documento OpenAPI*
aa-registry	string (url)	OBBLIGATORIO. URI presso il registro SPID delle AA dove è esposto il presente *documento OpenAPI*

[1] Fielding, Roy Thomas (2000). "Representational State Transfer (REST)". Architectural *Styles and the Design of Network-based Software Architectures* (PhD). University of California, Irvine

[2] Modello di Interoperabilità - Linee guida sul Modello di Interoperabilità pubblicate da AgID e relative regole tecniche

[3] Raccomandazioni Agid TLS e CipherSuite (https://www.agid.gov.it/it/sicurezza/tls-e-cipher-suite)

Table 1: OpenAPI - estensione x-spid

Viene di seguito riportato un esempio di documento OpenAPI con la specifica dell'elemento

```
openapi: 3.0.2
info:
 title: Esempio di Attribute Authority
 description: "Descrizione delle informazioni restituite dall\u2019 Attribute Authority"
 termsOfService: https://api.aa.it/tos
 contact:
  name: Contatto di riferimento
  url: https://www.aa.it
 version: 1.1.0
 x-spid:
  aa-home: https://api.aa.it/v1.0.0/index.json
  aa-registry: ' https://registry.spid.gov.it/metadata/aa/ente.json '
  aa-version: 1.0.0
```

Esempio 1: OpenAPI - estensione x-spid

Viene di seguito riportato un esempio di *path* e *operation,* il primo definisce quali sono gli endpoint (risorse) che l'API espone mentre le operazioni sono i metodi HTTP utilizzati per manipolare questi percorsi, come ad esempio GET.

```
paths:
 /api/v1.1.1/qualifica:
  get:
   operationId: getQualifica
   summary: |
     Esempio di risorsa con Autenticazione di tipo UserConsent.
   security:
    - UserConsent: [get-qualifica, ]
```

*Esempio 2: Risorsa protetta da autorizzazione con **scope** get-qualifica.*

2.2 Tipi di dato

L'AA descrive nel proprio *documento OpenAPI* il *tipo* di dato di ogni parametro accettato in input e di ogni dato restituito in output in conformità con quanto indicato nel MODI e specificando, ove possibile, gli ulteriori vincoli di formato e sintassi associati ai campi. Le AA possono utilizzare il campo *externalDocs* per inserire il riferimento specifico ad un vocabolario controllato o ad una ontologia al fine di definire rispettivamente la sintassi o la semantica propria del dato.

2.3 Servizio di acquisizione del consenso (Authorization Service)

Nel *documento OpenAPI* devono essere presenti i riferimenti al Token Endpoint e all'Authorization Endpoint specificati usando l'apposito SecurityScheme Object definito in OpenAPI.

```
components:
  securitySchemes:
    UserConsent:
      type: openIdConnect
      openIdConnectUrl: https://aa.it/.well-known/openid-configuration
      description: |-
        Questa sezione esplicita il riferimento all'URL di configurazione
        di OIDC relativo al provider. Deve contenere le informazioni
        indicate in:
        https://openid.net/specs/openid-connect-discovery-1_0.html
```

Esempio 3: La specifica del meccanismo di autenticazione basato su OIDC e denonimato UserConsent

Il documento ritornato dall'openid-configuration DEVE contenere i riferimenti agli endpoint "authorization" e "token". Affinché l'AA possa ottenere il consenso dell'utente il SP deve redirezionare l'utente presso tale servizio. Il servizio di acquisizione del consenso, prevedendo l'interazione con l'utente, non può essere *consumato* in modalità "machine to machine". Inserendo il riferimento ad *authorization* e *token endpoint* nei *#/components/securitySchemes* viene identificato Il servizio di acquisizione del consenso diversamente dagli altri servizi dell'interfaccia.

Di seguito un contenuto di esempio dei metadati degli endpoint OIDC usati per ottenere il consenso:

```
{
  "version": "3.0",
  "issuer": "https://127.0.0.1:5000",
  "token_endpoint_auth_methods_supported": [
    "private_key_jwt"
  ],
  "claims_parameter_supported": true,
  "grant_types_supported": [
    "authorization_code",
    "urn:ietf:params:oauth:grant-type:jwt-bearer",
    "refresh_token"
  ],
  "subject_types_supported": [
    "public",
```

```
  "pairwise"
],
"introspection_endpoint": "https://127.0.0.1:5000/introspection",
"response_types_supported": [
  "code",
],
"response_modes_supported": [
  "query",
  "fragment",
  "form_post"
],
"request_object_signing_alg_values_supported": [
  "RS256",
  "RS384",
  "RS512",
  ...
],
"request_object_encryption_alg_values_supported": [
  "RSA-OAEP",
  "RSA-OAEP-256",
  ...
],
"request_object_encryption_enc_values_supported": [
  "A128GCM",
  "A192GCM",
  "A256GCM",
  ....
],
"claim_types_supported": [
  "normal",
  "aggregated",
  "distributed"
],
"authorization_endpoint": "https://127.0.0.1:5000/authorization",
"token_endpoint_auth_signing_alg_values_supported": [
  "RS256",
   ...
],
"token_endpoint": "https://127.0.0.1:5000/token",
"userinfo_signing_alg_values_supported": [
  "RS256",
  ...
],
"client_authn_method": [
  "bearer_header"
],
"end_session_endpoint": "https://127.0.0.1:5000/session",
```

```
 "acr_values_supported": [
  "https://www.spid.gov.it/SpidL1",
  "https://www.spid.gov.it/SpidL2"
 ],
 "jwks_uri": "https://127.0.0.1:5000/static/jwks.json",
 "id_token_signing_alg_values_supported": [
  "RS256",
  ...
 ],
 "id_token_encryption_alg_values_supported": [
  "RSA-OAEP",
  "RSA-OAEP-256",
   ...
 ],
 "id_token_encryption_enc_values_supported": [
  "A128GCM",
  "A192GCM",
  "A256GCM",
   ...
 ],
 "scopes_supported": [
  "offline_access",
  "openid",
  "get-qualifica",
 ],
 "claims_supported": [
  "iss",
  "zoneinfo",
  "updated_at",
  "sub",
 ]
}
```


Esempio 4: il documento openid-configuration che descrive le caratteristiche del provider di autenticazione OIDC

Se una chiamata all'API per la richiesta di attributi qualificati non può essere soddisfatta perché manca il relativo consenso, l'AA DEVE ritornare una Response con codice di stato *HTTP 403 Forbidden* e un Contenuto di tipo json con l'indicazione descrittiva "*Consent Required*" all'interno dell'attributo **error_description**.
Nella Figura di seguito vengono illustrate le interazioni tra le entità coinvolte all'interno del flusso quando una autenticazione utente con relativo rilascio del consenso sono richieste. È importante considerare che una AA (Attribute Authority) e un AS (Authorization Server) possono corrispondere ad un'unica entità oppure ad entità diverse.

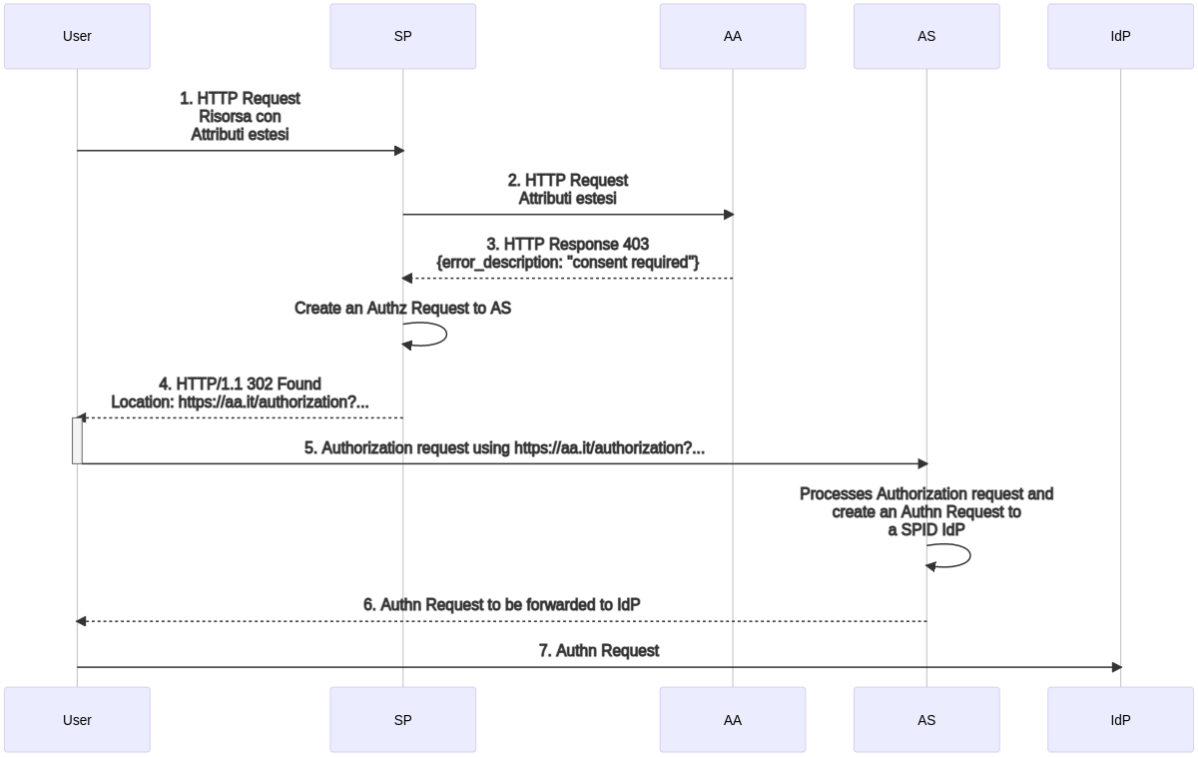

Figura 1: Sessione di richiesta Attributi con richiesta di autenticazione. Gli endpoint di AA e AS vengono erogati da entità diverse.

2.4 Operazioni consentite

L' API DEVE esporre esclusivamente endpoint che realizzano operazioni di interrogazione verso l'AA al fine di ottenere gli attributi qualificati dell'utente o i riferimenti degli utenti che soddisfano i criteri di ricerca specificati (esclusivamente nei casi in cui non è previsto il consenso diretto). I formati dei nomi utilizzati per gli endpoint e i verbi HTTP sui quali sono esposte le operazioni DEVONO essere conformi agli standard HTTP e alle indicazioni contenute nel MODI[2].

2.5 Versioning

La specifica OpenAPI DEVE contenere nel campo *"#/info/version"* l'indicazione della versione secondo il formato: MAJOR.MINOR.PATCH conformemente alle indicazioni del MODI.
I numeri MAJOR, MINOR e PATCH sono da intendere secondo l'accezione SemVer[4] , quindi le eventuali modifiche alle API devono permettere evoluzioni compatibili nel tempo.

[4] Semantic Version Specification. (https://semver.org/)

Nota:	<HOST> è il dominio dell'AA <MAJOR> è il numero MAJOR corrispondente alla versione dell'API <MINOR> è il numero MINOR corrispondente alla versione dell'API <PATCH> è il numero PATCH corrispondente alla versione dell'API

Esempio:

```
openapi: 3.0.2
info:
  title: Esempio di Attribute Authority
  version: 1.4.43
```

Esempio 5: OpenAPI - versioning

Gli endpoint DEVONO supportare il versionamento sull'URL. Sarà possibile quindi effettuare una interrogazione verso una specifica versione dell'API indicando l'URL corrispondente, che DEVE essere formata nel seguente modo:

*https://**<HOST>**/api/v**<MAJOR>[.<MINOR>[.<PATCH>]]**/**<operation>***

Nota:	<HOST> è il dominio dell'AA <MAJOR> è il numero MAJOR corrispondente alla versione dell'API <MINOR> è il numero MINOR corrispondente alla versione dell'API <PATCH> è il numero PATCH corrispondente alla versione dell'API <operation> è il path della chiamata API

I numeri indicati tra parentesi quadre non sono obbligatori e possono essere omessi. In tal caso verrà indirizzata la versione corrispondente al numero MINOR o PATCH, rispettivamente, più alto disponibile. Sono di seguito riportati alcuni esempi di URI per versioni differenti della stessa chiamata API.

```
https://aa.it/api/v1/qualifica
https://aa.it/api/v1.1/qualifica
```

Esempio 6: OpenAPI - versioning URL

Per realizzare il versionamento sull'URL, l'attributo *url* di ognuno degli elementi presenti all'interno della sezione *"#/servers"* del *documento Open API*, DEVE includere il numero di versione coerentemente con quanto indicato nell'elemento *"#/info/version"*.

```
...
servers:
- "url": https://aa.it/v1
 ...
```

Esempio 7: OpenAPI - versioning e servers

2.6 Documentazione API

L'AA deve esporre la documentazione API generata automaticamente sulla base del *documento OpenAPI* su un URL pubblica afferente al dominio della stessa AA espresso nella seguente forma, coerentemente con quanto già indicato nel paragrafo **2.5 Versioning**.

*https://**<HOST>**/api/v**<MAJOR>[.<MINOR>[.<PATCH>]]***

2.7 Raggruppamenti per tipologie di accesso ai dati

L'accesso ai dati, per ogni operazione, può essere classificato sulla base delle seguenti casistiche:

a) "public" - Il dato è open, di pubblico dominio o liberamente accessibile. In tal caso l'accesso al dato non richiede l'acquisizione del consenso da parte dell'utente.
b) "protected" - L'accesso al dato è riservato alle pubbliche amministrazioni o ai SP che hanno una specifica convenzione con l'AA. In tal caso se la richiesta proviene da una pubblica amministrazione o da un SP che ha una convenzione con l'AA per accedere liberamente al dato, l'accesso non richiede l'acquisizione del consenso. In tutti gli altri casi, invece, è lasciata facoltà all'AA di decidere se l'acquisizione del consenso è necessaria.
c) "private" - L'accesso al dato è consentito solo previa acquisizione del consenso. Quest'ultima casistica non si applica quando il richiedente è una pubblica amministrazione.

Al fine di descrivere in maniera chiara le operazioni soggette all'acquisizione del consenso, il campo *"#/tags"* DOVREBBE contenere almeno i seguenti *Tag* utili a raggruppare le corrispondenti operazioni:

name	description
public	Accesso pubblico
protected	Accesso pubblico per pubbliche amministrazioni e SP convenzionati
private	Accesso previa acquisizione di consenso dell'utente

Table 2: OpenAPI - tags per tipologie di accesso ai dati

Detti tag, qualora siano presenti nel *documento OpenAPI*, conterranno il campo externalDocs con le seguenti informazioni:

description	url
Linee Guida Attribute Authority SPID	*Il riferimento presso il sito AgID dove sono pubblicate le Linee Guida Attribute Authority SPID*

Table 3: OpenAPI - externalDocs per tags predefiniti

A solo titolo esemplificativo, viene di seguito riportata la parte del *documento OpenAPI* che definisce i suddetti Tag.

```
tags:
- description: Accesso pubblico
  externalDocs:
    description: Linee Guida Attribute Authority SPID
    url: https://www.agid.gov.it/...
  name: public
- description: Accesso pubblico per pubbliche amministrazioni e SP convenzionati
  externalDocs:
    description: Linee Guida Attribute Authority SPID
    url: https://www.agid.gov.it/...
  name: protected
- description: "Accesso previa acquisizione di consenso dell'utente"
  externalDocs:
    description: Linee Guida Attribute Authority SPID
    url: https://www.agid.gov.it/...
  name: private
```

Esempio 8: OpenAPI - tags per tipologie di accesso al dato

L'AA può indicare ulteriori raggruppamenti di operazioni tramite "*Tag*" per descrivere, ad esempio, quali operazioni sono consentite a determinate pubbliche amministrazioni o SP senza la necessità di acquisizione del consenso sulla base di una norma giuridica, regolamento o convenzione. In tal caso devono essere considerate le seguenti indicazioni:

- il valore del campo *name* del Tag deve riportare come prefisso *"protected-"* seguito dal nome identificativo del Tag, senza spazi.
- nell'elemento *externalDocs* devono essere specificati i campi *description* e *url* in riferimento alla specifica norma, regolamento o convenzione che giustifica la politica di accesso.

```
tags:
- description: Accesso pubblico per CAF convenzionati
  externalDocs:
    description: "Convenzione che regola l'accesso ai dati per i CAF"
    url: https://...
  name: protected-convenzione-caf
```

Esempio 9: OpenAPI - tags per ulteriori raggruppamenti

2.8 Documentazione operazioni

Il campo *summary* di ogni oggetto *Operation* DEVE contenere una breve descrizione degli attributi qualificati che l'operazione permette di recuperare.

Il campo *description* di ogni oggetto *Operation* PUÒ indicare le modalità richieste per l'accesso al dato ed in particolare:

- se l'attributo richiede la raccolta esplicita del consenso dell'utente o meno;
- se è richiesta un'autenticazione SPID di un livello minimo;
- se sono consentite richieste continuative e la durata massima consentita.

È prevista per l'oggetto *Operation*, l'estensione *x-spid-operation*, con i campi indicati nella seguente tabella, per descrivere in maniera strutturata le caratteristiche elencate sopra.

campo	tipo	descrizione
consentRequired	string	Se = 'N' l'operazione non richiede il consenso Se = 'AA' l'operazione richiede il consenso diretto presso l'AA Se = 'SP' l'operazione richiede il consenso acquisito dal SP Se = 'AAorSP' l'operazione richiede il consenso diretto presso l'AA o acquisito dal SP
spidLevel	string	ACR del livello SPID minimo richiesto
offlineAccess	boolean	se true sono consentite richieste continuative
offlineAccessDurationMax	integer	num. max di secondi consentiti per richieste continuative (es. 10giorni = 864000)

Table 4: OpenAPI - estensione x-spid-operation

2.9 Autenticazione

Gli endpoint delle API per la richiesta di attributi qualificati, ad eccezione degli endpoint ad accesso pubblico (Tipologia di accesso "public"[5]), devono essere protetti tramite autenticazione.

2.9.1 Autenticazione senza necessità di consenso dell'utente

[5] Vedi paragrafo 2.7 Raggruppamenti per tipologie di accesso ai dati

L'unica tipologia di autenticazione consentita è la *Bearer Authentication*[6] tramite token in formato *JWT*[7]. DEVONO essere seguite le indicazioni di sicurezza contenute nel MODI[2] e in RFC8725[8]. La struttura dei token JWT nelle interazioni che non richiedono il consenso dell'utente si basano sui pattern di autenticazione tramite JWT descritti dal MODI e DEVONO contenere **almeno** i campi indicati di seguito.

Nell'header, i parametri:

Parametro	Descrizione
typ	valorizzato con la stringa "JOSE" ad indicare che il token è un JWT (JWS o JWE) secondo le specifiche JWS Compact Serialization o JWE Compact Serialization.
alg	valorizzato con l'identificativo JWA dell'algoritmo crittografico utilizzato per l'autenticazione JWE.
x5c	valorizzato con il certificato o la catena dei certificati, in formato X.509, corrispondente alla chiave pubblica del certificato di sigillo elettronico utilizzato. Il certificato o la catena dei certificati sono codificati come array JSON di stringhe corrispondenti ai valori dei certificati in formato DER. Ogni stringa nell'array è codificata in Base64. Il certificato contenente la chiave pubblica del sigillo utilizzato per firmare il token deve essere la prima stringa dell'array.
x5t#S256	valorizzato con l'hash del certificato corrispondente alla chiave pubblica del certificato di sigillo elettronico utilizzato. Il destinatario del token deve poter recuperare il certificato e validarlo.

Table 5: claim della intestazione del JWT

Nel payload in formato JWT, almeno i claim:

Claim	Descrizione
sub	Identificatore del mittente del messaggio (SP o AA) o del soggetto che richiede l'autenticazione che lo contraddistingue univocamente nella federazione.
iss	Identificatore del soggetto che ha generato e firmato il token. Se il token è generato dal mittente il valore corrisponde a quello del claim "sub".
aud	Identificatore del destinatario del messaggio (AA o SP) che lo contraddistingue univocamente nella federazione
iat	Data/ora di emissione del token in formato UNIX Timestamp
exp	Data/ora di scadenza del token in formato UNIX Timestamp.
jti	Identificatore unico del token che è possibile utilizzare per prevenirne il riuso illecito. Deve essere di difficile individuazione da parte di un attaccante e composto da una stringa casuale

Table 6: claim del payload del JWT

Lo schema di autenticazione predefinito, valido per tutti gli endpoint, deve essere dichiarato nell'elemento *"#/components/securitySchemes"* e referenziato nell'elemento *security* dell'*Operation*. A titolo esemplificativo, viene di seguito riportata la parte del *documento OpenAPI* che definisce lo schema di autenticazione predefinito.

[6] Bearer Token Authentication. (https://tools.ietf.org/html/rfc6750)

[7] JSON Web Token (JWT). (https://tools.ietf.org/html/rfc7519)

[8] JSON Web Token Best Current Practices (https://tools.ietf.org/html/rfc8725)

```
components:
  ...
  securitySchemes:
    SPID Attribute Authority Security Scheme:
      bearerFormat: JWT
      description: JWT Token Authentication Scheme
      scheme: bearer
      type: http
   ...
security:
- SPID Attribute Authority Security Scheme: []
...
```

Esempio 10: OpenAPI - autenticazione

2.9.2 Autenticazione con consenso dell'utente

L'autenticazione con consenso dell'utente è descritta nella sezione 2.3 Servizio di acquisizione del consenso.

2.10 Response

Le Response a richieste di attributi qualificati potranno essere restituite sia come Media-Type *application/json* che come Media-Type *application/jwt*. Il documento OpenAPI deve comunque indicare sempre lo schema del payload dell'oggetto JSON Web Token restituito. Nel caso di Media-Type *application/jwt*, infatti, il body della Response conterrà un JSON Web Token, realizzato secondo le specifiche indicate nelle presenti linee guida (**4.2 Attribute Response**), il cui payload dovrà essere un oggetto JSON dello stesso schema dichiarato per il caso di Media-Type *application/json*.

```
"200": {
   "description": "Successful",
   "content": {
      "application/jwt": {
         "schema": {
            "type": "string"
         }
      },
      "application/json": {
         "schema": { "$ref": "#/components/schemas/SuccessfulResponse" }
      }
   }
}
```

Esempio 11: OpenAPI - response MediaType

2.11 Errori

Se la richiesta di attributi qualificati non può essere soddisfatta, l'AA DEVE restituire una Response, secondo le indicazioni riportate in **2.10 Response**, conforme al MODI[2] e basate sull' RFC7807[9]:

element	descrizione
status	HTTP status code
type	URI che rimanda al codice specifico dell'errore, definito dall'AA
title	Descrizione standard dell'errore
detail	Messaggio specifico dell'errore

Table 7: OpenAPI - errori Response

Gli elementi type e detail possono essere utilizzati dall'AA per specificare ulteriormente l'errore al fine di evidenziare casistiche di propria pertinenza. Nel caso in cui non esista una ulteriore specificazione essi potranno essere omessi. Sono di seguito riportati alcuni esempi di risposta di errore.

```
HTTP/1.1 401 Unauthorized
{
   "status": 401,
   "title": "Autenticazione fallita"
}
```

Esempio 12: OpenAPI - errore di Autenticazione fallita

```
HTTP/1.1 500 Internal Server Error
{
   "status": 500,
   "type": "https://aa.it/error/auth-failed",
   "title": "Internal Server Error",
   "detail": "Messaggio di errore personalizzato"
}
```

Esempio 13: OpenAPI - errore 500 Personalizzato

3. Richiesta di attributi (Attribute Request)

Sono di seguito presentati i flussi che descrivono i possibili scenari relativi alla richiesta di attributi.

[9] RFC7807 - Problem Details for HTTP APIs (https://tools.ietf.org/rfc/rfc7807.txt)

3.1 Richiesta senza necessità del consenso

Viene di seguito descritto il flusso valido nel caso in cui non sia necessaria l'acquisizione del consenso dell'utente alla trasmissione dei dati. Questo flusso rientra nella casistica standard del modello di interoperabilità.

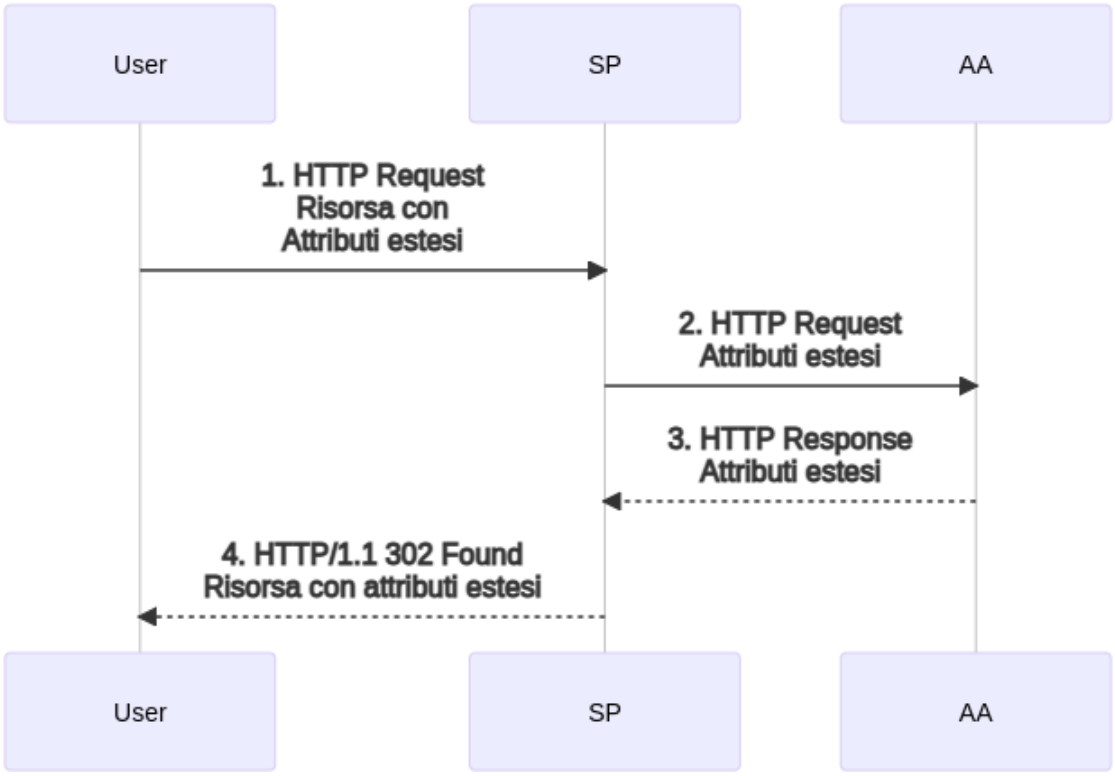

Figura 2: Flusso - richiesta senza necessità del consenso per risorsa di tipo public

1. L'utente chiede di utilizzare un servizio presso il SP (Service Request) per il quale è necessario conoscere uno o più attributi qualificati.

2. Il SP, avendo necessità di conoscere gli attributi qualificati per erogare il servizio esegue una chiamata API alla specifica AA che ha la facoltà di attestare e restituire tali attributi qualificati.

3. L'AA verifica l'integrità e l'autenticità della request (Attribute Request) e, dopo aver accertato la possibilità di rispondere alla richiesta senza che sia necessario ottenere il consenso da parte dell'utente, restituisce al SP gli attributi richiesti.

4. Il SP verifica l'integrità e l'autenticità della response (Attribute Response) ottenuta dall'AA e utilizza gli attributi qualificati ricevuti per erogare il servizio all'utente (Service Response).

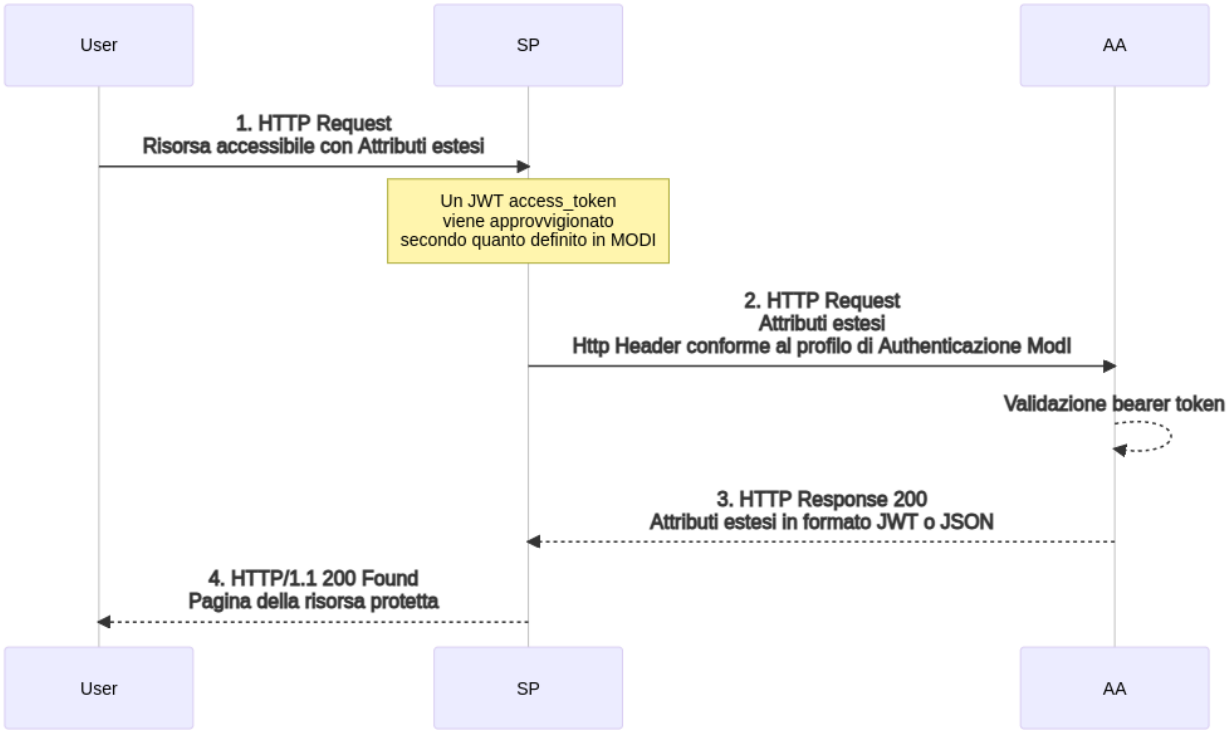

Figura 3: Esempio di Richiesta senza consenso utente, flusso "machine to machine" secondo il modello di interoperabilità (MODI).

3.2 Richiesta con necessità del consenso e riautenticazione

Viene di seguito descritto il flusso nel caso in cui l'acquisizione del consenso dell'utente alla trasmissione dei dati sia necessaria. Tale acquisizione avviene tramite autenticazione dell'utente presso l'IdP SPID.

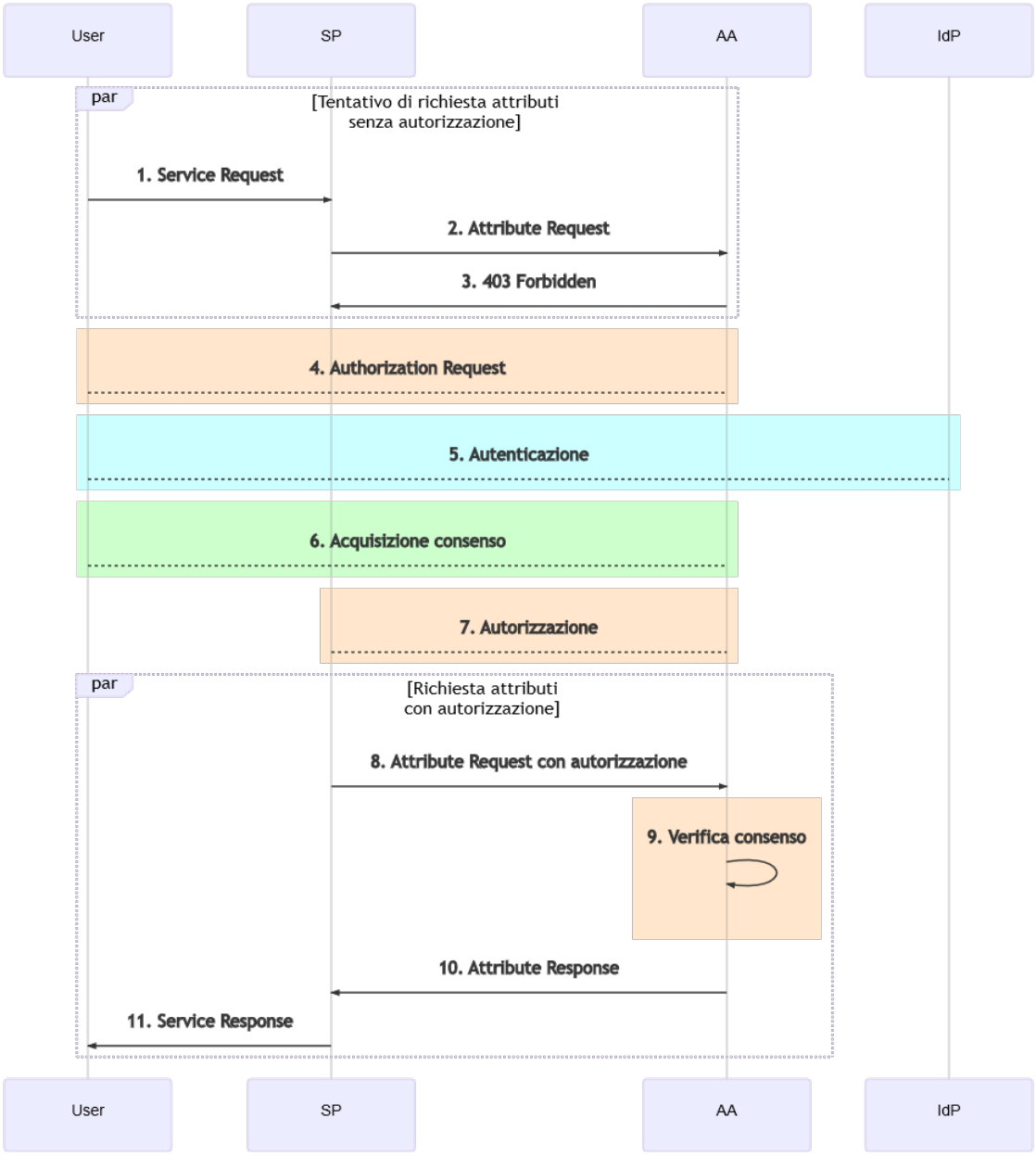

Figura 4: Flusso - richiesta con necessità del consenso e riautenticazione (semplificato)

Il rilascio dell'autorizzazione dell'utente al SP per effettuare le richieste di attributi è implementato tramite il flusso standard di *Authorization Code Flow* definito dai protocolli OAuth2.0[10] e OIDC[11]. Il flusso è mostrato nello schema seguente che riprende parte del flusso

[10] The OAuth 2.0 Authorization Framework (https://tools.ietf.org/rfc/rfc6749.txt)

[11] OIDC Core 1.0 (https://openid.net/specs/openid-connect-core-1_0.html)

descritto nel paragrafo **3.2 Richiesta con necessità del consenso e riautenticazione**. Per continuità di lettura, lo schema seguente riprende anche la stessa numerazione, specificando nel dettaglio i passi relativi ai processi di autorizzazione, autenticazione e acquisizione del consenso. A differenza del precedente schema, l'AA è stavolta divisa rispetto all'**Authorization Server (AS)**.

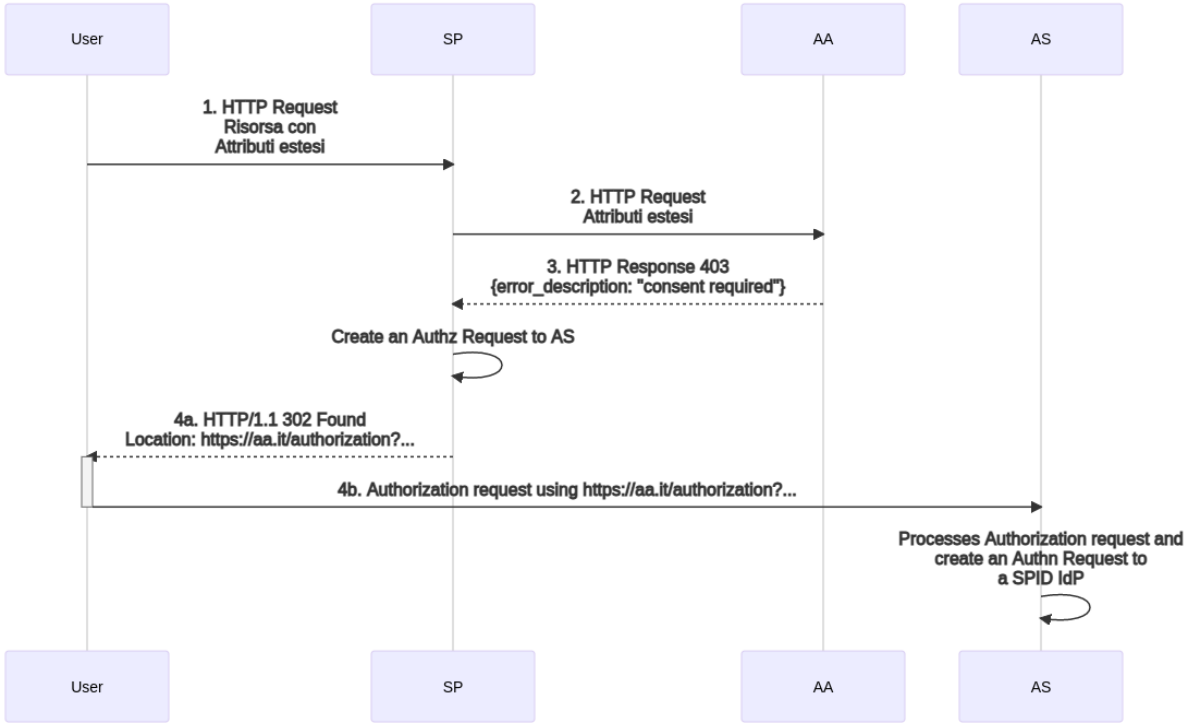

Figura 5: Flusso - richiesta con necessità del consenso e riautenticazione

1. L'utente chiede di utilizzare un servizio presso il SP (**Service Request**) per il quale è necessario conoscere uno o più attributi qualificati.

2. Il SP avendo necessità di conoscere gli attributi qualificati per erogare il servizio esegue una chiamata API alla specifica AA che ha la facoltà di attestare e restituire tali attributi qualificati.

3. L'AA verifica l'integrità e l'autenticità della request (**Attribute Request**) e, avendo necessità di ottenere il consenso per restituire gli attributi richiesti, restituisce al SP un messaggio del tipo 403 Forbidden, contenente un oggetto JSON con i parametri **error_description** valorizzato a "Consent Required".

4.

a. Il SP valida la response ricevuta dalla AA e costruisce una **Authorization Request**
b. Il SP reindirizza l'utente sull'AS con l'URL di **Authorization Request**

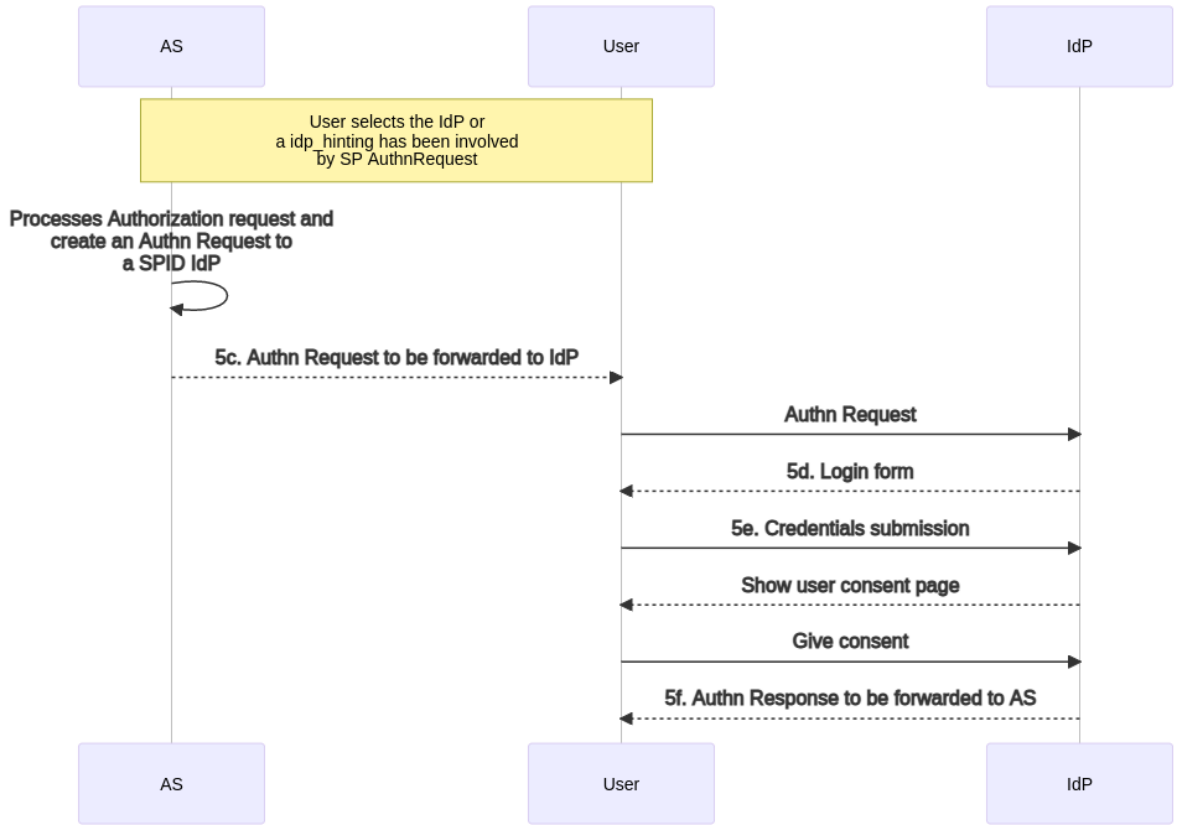

Figura 6: Flusso - Autenticazione richiesta dalla AS presso l'IdP SPID per conto dell'utente da identificare

5.

a. L'AS presenta all'utente la scelta dell'IdP presso il quale effettuare l'autenticazione ai fini dell'acquisizione del consenso (se idp hinting fosse assente)
b. L'utente seleziona l'IdP (se idp hinting fosse assente)
c. L'AS reindirizza l'utente presso l'IdP selezionato con la SAML Request SPID.
d. L'IdP presenta all'utente il form di login
e. L'utente si autentica presso l'IdP
f. L'IdP reindirizza il flusso utente verso L'AS con la SAML Response SPID contenente gli attributi dell'utente necessari a verificare se è possibile concedere l'autorizzazione alla richiesta di attributi qualificati

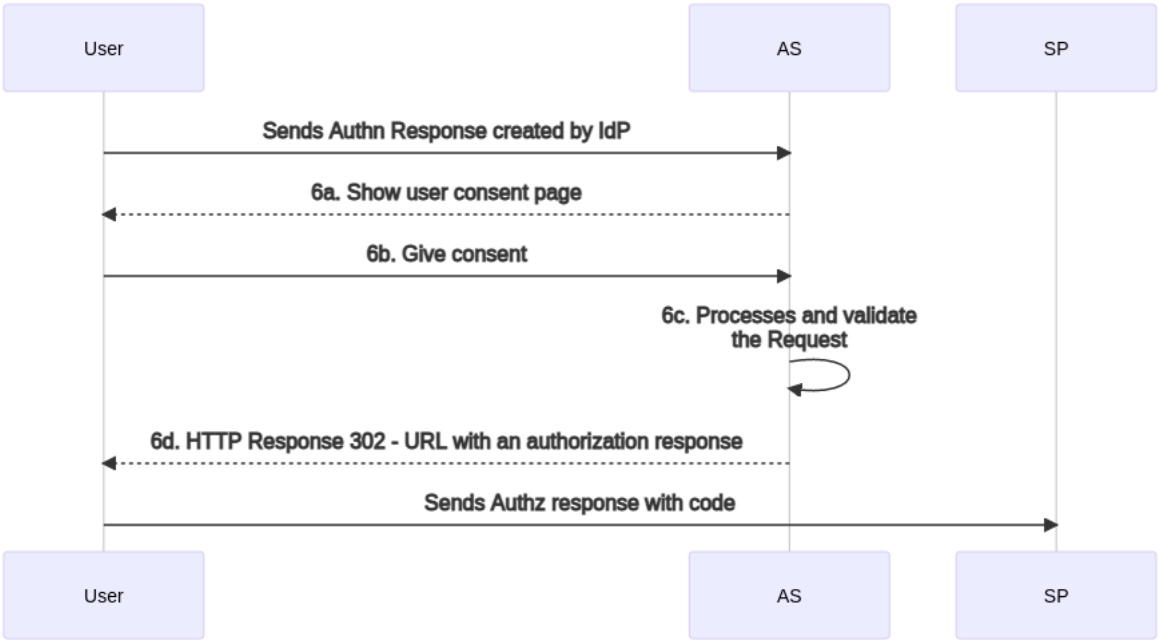

Figura 7: Flusso -Identificazione dell'utente presso l'AS e rilascio del codice di autorizzazione al SP

6.

a. L'AS chiede all'utente il consenso alla trasmissione degli attributi qualificati al SP
b. L'utente fornisce il consenso alla trasmissione degli attributi qualificati al SP
c. L'AS verifica I parametri forniti dal SP (richiesta API, attributi richiesti, identità del SP, contesto o finalità dichiarata dal SP) con l'**Authorization Request** (punto 4), gli attributi dell'utente rilasciati dall'IdP (soggetto interessato) e se l'utente ha fornito il consenso al rilascio degli attributi qualificati (punto 6)
d. L'AS reindirizza l'utente presso il SP con l'URL di **Authorization Code**

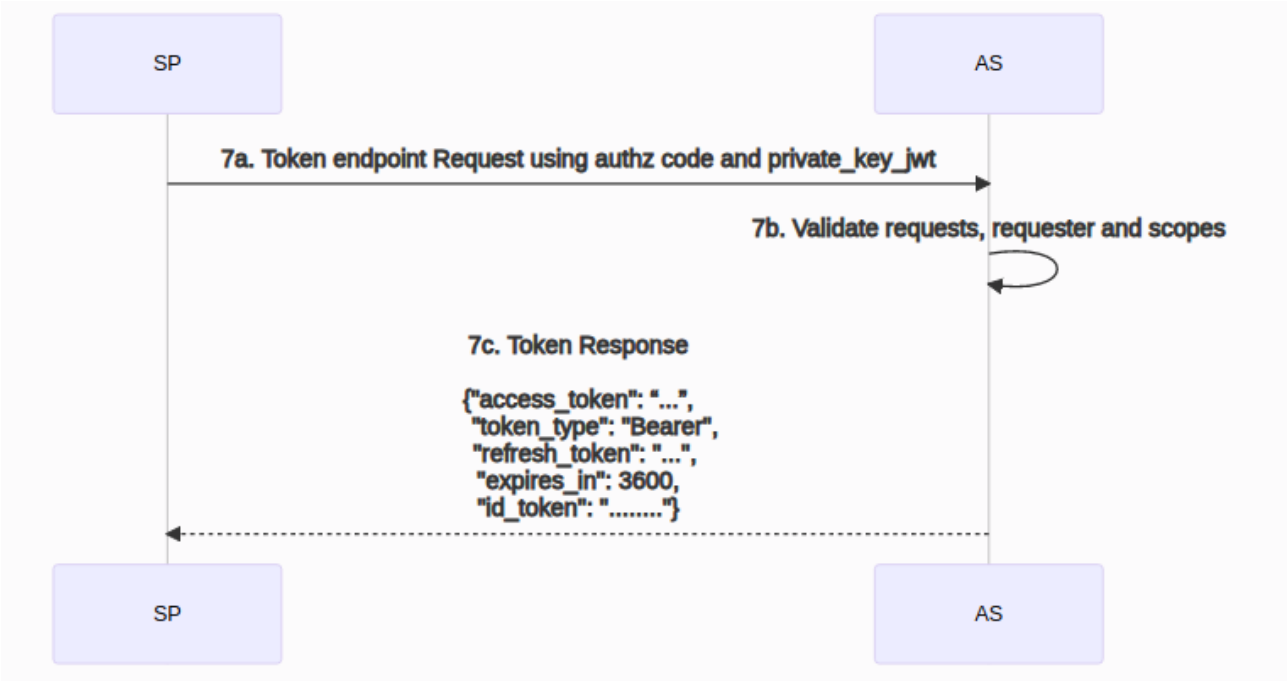

Figura 8: Flusso - Rilascio dell'Access Token al SP per il consumo delle risorse di AA

7.
   a. Il SP effettua una richiesta verso l'endpoint /token presso l'AS presentando l'**Authorization Code** al fine di ottenere un **Access Token** valido
   b. L'AS verifica la validità dell'Authorization Code
   c. L'AS fornisce un Access Token valido

8. Il SP effettua nuovamente la chiamata API (**Attribute Request**) alla AA per ottenere gli attributi qualificati autenticandosi tramite l'**Access Token** ottenuto dall'AS (punto 7) a seguito dell'autorizzazione.

9.
   a. La AA verifica la firma e la validità dell'Access Token, infine per quali scopi questo sia stato rilasciato
   b. la AA restituisce al SP gli attributi richiesti.

10. Il SP verifica l'integrità e l'autenticità della response (**Attribute Response**) ottenuta dalla AA e utilizza gli attributi qualificati ricevuti per erogare il servizio all'utente (**Service Response**).

3.2.1 Suggerimento dell'IdP da utilizzare (IdP hinting)

È RACCOMANDATO che il SP e la AA implementino il supporto di idp hinting. IdP hinting è un meccanismo utilizzato per trasmettere alla AA un suggerimento sull'IdP da utilizzare per autenticare l'utente.

Il suggerimento dell'IdP può snellire notevolmente il processo di autenticazione da parte dell'utente prevenendo che lo SPID Button venga nuovamente presentato.

IdP hinting è una facilitazione che DOVREBBE essere implementata dalla AA per migliorare l'esperienza utente, in caso in cui il SP non lo supporti la AA DEVE comunque implementare e mostrare all'utente la schermata contenente lo SPID button.

Per usare questa funzionalità il SP POTRÀ includere, all'interno dell'url di redirect per la richiesta di una Autorizzazione presso la AA, il parametro idphint valorizzato con un array di valori *url encoded*, come di seguito esemplificato:

idphint=['https%253A%252F%252Fidp.spid.it']

Il valore definito in idphint DEVE corrispondere all'IdP presso il quale l'utente è stato autenticato precedentemente, al primo accesso presso il SP richiedente.

3.3 Richieste continuative

I flussi relativi a richieste continuative ricadono negli schemi di flusso presentati nei paragrafi precedenti, **3.1 Richiesta senza necessità del consenso** e **3.2 Richiesta con necessità del consenso e riautenticazione**.

Il flusso di richiesta continuativa nel caso di pubbliche amministrazioni o SP per i quali esiste specifica convenzione con l'AA corrisponde al flusso presentato al paragrafo **3.1 Richiesta senza necessità del consenso**. Il flusso di richiesta continuativa con necessità del consenso e riautenticazione corrisponde al flusso presentato al paragrafo **3.2 Richiesta con necessità del consenso e riautenticazione** con le seguenti differenze:

- L'authorization request che il SP invia all'AA per ottenere il consenso, deve specificare che l'autorizzazione richiede di effettuare richieste continuative (**offline_access** scope).

- L'autorizzazione rilasciata dall'AA potrà essere rinnovata, senza la necessità di far ripetere l'autenticazione all'utente, per effettuare ulteriori richieste di attributi. Il rinnovo dell'autorizzazione può essere richiesto un numero illimitato di volte per il tempo massimo concordato per la specifica richiesta continuativa. Per maggiori dettagli in merito all'autorizzazione per richieste continuative fare riferimento al paragrafo **5 Autorizzazione per richieste continuative**.

3.4 Attribute Request

Per richiedere gli attributi qualificati, il SP invia una Attribute Request al particolare endpoint della specifica AA che detiene il dato. Le modalità di interrogazione sono definite nella documentazione OpenAPI esposta dalla AA. L'attribute request deve contenere le credenziali in formato Bearer Token JWT con le quali autenticarsi presso l'endpoint se questo non fosse di pubblico accesso (Tipologia di accesso "public"[12]). L'autenticazione deve essere conforme alle indicazioni contenute nei paragrafi **2.9 Autenticazione** e **6. Intestazioni e Payload comuni dei JWT**.

```
GET /v1.0.0/attribute HTTP/1.1
Host: aa.it
Authorization: Bearer eyJ0eXAiOiJKT1NFIiwiYWxnIjoiSFMyNTYiLCJlbmMiOiJIUzI1NiIsIng1YyI6IkNlc ...
```

Esempio 14: Intestazione di una Attribute Request con credenziali di autenticazione di tipo Bearer

3.5 Attribute Response

In risposta ad una Attribute Request l'AA ritorna un messaggio che può contenere gli attributi qualificati richiesti oppure l'indicazione dell'errore riscontrato per il quale non è possibile trasmettere gli attributi richiesti. I possibili dati contenuti nella risposta o le specifiche dell'errore sono definiti nella documentazione OpenAPI dell'AA.

La risposta DOVREBBE avere Media-Type *application/jwt* in modo da poter firmare e cifrare il contenuto, ma PUÒ anche avere Media-Type *application/json*. Per maggiori dettagli in merito al JWT fare riferimento al paragrafo **6. Intestazioni e Payload comuni dei JWT**.

```
HTTP/1.1 200 OK
```

[12] Vedi paragrafo 2.7 Raggruppamenti per tipologie di accesso ai dati

```
Content-Type: application/jwt
eyJ0eXAiOiJKT1NFIiwiYWxnIjoiSFMyNTYiLCJlbmMiOiJIUzI1NiIsIng1YyI6IkNlcnRpZmljYXRvL2NvZGlmaWNhd ...
```

Esempio 15: Attribute Response con JWT firmato (JWS)

4. Dettagli tecnici delle Request e delle Response

In questa sezione sono descritte le specifiche delle Richieste di Autorizzazione e conseguente ottenimento del Token di autorizzazione per il consumo delle risorse presso le AA.

Sono di seguito riportati i parametri comuni presenti nell'intestazione dei JWT.

Parametro	Descrizione
typ	valorizzato con la stringa "JOSE" ad indicare che il token è un JWT (JWS o JWE) secondo le specifiche JWS Compact Serialization o JWE Compact Serialization.
alg	valorizzato con l'identificativo JWA dell'algoritmo crittografico utilizzato per l'autenticazione JWE.
enc	valorizzato con l'identificativo JWA dell'algoritmo crittografico utilizzato per la cifratura del pacchetto JWE.
x5c	valorizzato con il certificato o la catena dei certificati, in formato X.509, corrispondente alla chiave pubblica del certificato di sigillo elettronico utilizzato. Il certificato o la catena dei certificati sono codificati come array JSON di stringhe corrispondenti ai valori dei certificati in formato DER. Ogni stringa nell'array è codificata in Base64. Il certificato contenente la chiave pubblica del sigillo utilizzato per firmare il token deve essere la prima stringa dell'array.

Table 8: Intestazioni comuni JWT

Un esempio di intestazione come sopra definita è:

```
{
    "typ" : "JOSE",
    "alg" : "ES256",
    "enc" : "A256GCM",
    "x5c" : "Certificato/codificato+Base64",
    "crit": ["x5c"]
}
```

Esempio 16: Intestazioni comuni JWT

Sono di seguito riportati i claim comuni, non caratterizzanti, utilizzati nel payload. Nel caso la struttura sia cifrata in un pacchetto JWE, tali claim dovranno essere presenti:

- sia nel payload cifrato del pacchetto JWE,
- che nella porzione non cifrata del pacchetto JWS.

Questa procedura, prevista dalla norma RFC-7519, serve sia ad irrobustire ulteriormente l'integrità del pacchetto, che a migliorare la convalida applicativa, mantenendo la confidenzialità relativamente ai dati personali dichiarati (p.es. durante l'archiviazione dei messaggi nei log di sistema).

Parametro	Descrizione
jti	Identificatore unico del token che è possibile utilizzare per prevenirne il riuso illecito. Deve essere di difficile individuazione da parte di un attaccante e composto da una stringa casuale
iss	Identificatore del mittente del messaggio (SP o AA) che lo contraddistingue univocamente nella federazione.
aud	Identificatore del destinatario del messaggio (AA o SP) che lo contraddistingue univocamente nella federazione
iat	Data/ora di emissione del token codificato come Unix timestamp
exp	Data/ora di scadenza del token codificato come Unix timestamp.

Table 9: Payload comune JWT

Il parametro exp rappresenta la data di scadenza del token, oltre questa data il token non sarà ritenuto più valido; tale orario è normalmente pari a iat più un valore di timeout pari a 5 minuti, fatta eccezione per le Authorization Response per richieste di attributi continuative (scope **offline_access**).

4.1 Authorization Request con Autenticazione e consenso utente

Per richiedere il consenso dell'utente e ottenere l'autorizzazione ad effettuare la richiesta di attributi presso la AA, il SP redireziona l'utente presso l'authorization endpoint. L'Authorization Request deve contenere i seguenti parametri, in conformità alle Linee Guida OpenID Connect in SPID[13], ponendo attenzione alle seguenti restrizioni, da considerarsi valide esclusivamente per il profilo AA:

1. La richiesta di autorizzazione NON DEVE avere tra gli **scope** l'attributo **profile**. In caso questa dovesse essere presente l'AS ignorerà questo scope specificando nella successiva Response di Token endpoint gli scopi per i quali l'access token è stato rilasciato, escludendo tra questi **profile**;
2. Il Client non consumerà Userinfo endpoint per ottenere il set esteso degli attributi dell'utente. Essendo lo scopo **profile** omesso oppure non autorizzato dall'AS, se il Client usasse l'access token per interrogare *Userinfo endpoint*, questo DOVRÀ rilasciare esclusivamente il claim *sub* relativo all'utente e nessun attributo a questo associabile;
3. Authorization code flow di OIDC viene utilizzato all'interno del modello di AA esclusivamente per l'acquisizione dell'access token, nessun claim relativo agli attributi di una identità digitale verranno inclusi all'interno dell'ID token.

[13] Linee Guida OpenID Connect in SPID (https://docs.italia.it/AgID/documenti-in-consultazione/lg-openidconnect-spid-docs)

Parametro	Tipo	Descrizione	Valori ammessi
client_id	Richiesto	URI che identifica univocamente il SP come da Registro SPID.	https://rp.spid.agid.gov.it deve contenere il FQDN dell'entità
response_type	Richiesto	Il tipo di credenziali che deve restituire l'OP.	code
scope	Richiesto	Ambito di utilizzo dell'autorizzazione	**offline_access** se specificato, identifica una richiesta di tipo continuativo **openid** **get-qualifica** (esempio) sulla base alla risorsa da interrogare presso la AA
redirect_uri	Richiesto	URI di reindirizzamento a cui verrà inviata la risposta	https://rp.spid.agid.gov.it/cb
code_challenge	Richiesto	Previene gli attacchi basati sulla intercettazione di authorization code	generato con algoritmo SHA256
code_challenge_method	Richiesto		S256
state	Richiesto	stringa opaca ad identificare univocamente, all'interno del RP, la sessione di autorizzazione	**fyZiOL9Lf2CeKuNT2JzxiLRDink0uPcd** (esempio)

Table 10: Parametri di OIDC authorization request

Di seguito un esempio della richiesta di Autenticazione:

GET
https://aa.it/authorization?client_id=https%253A%252F%252Frp.spid.agid.gov.it&code_challenge=qWJlMe0xdbXrKxTm72EpH659bUxAxw80&code_challenge_method=S256&redirect_uri=https%253A%252F%252Frp.spid.agid.gov.it%252Fcallback1%252F&response_type=code&scope=**openid**%20**offline_access**%20**get-qualifica**&state=fyZiOL9Lf2CeKuNT2JzxiLRDink0uPcd&***idphint****=%5B%27https%253A%252F%252Fidp.spid.it%27%5D*

Esempio 17: Authorization Request per richiesta di tipo non continuativo verso l'operazione /qualifica

4.2 Authorization Response

Alla ricezione di una Authorization Request, l'AS, dopo aver effettuato le necessarie verifiche e aver acquisito il consenso, crea una ulteriore Authentication Request presso l'IdP SPID per l'identificazione dell'utente. L'esito positivo di questa richiesta, accertata dall'avvenuta

autenticazione dell'utente presso l'IdP, consentirà il rilascio dell'**authorization code** da inviare al SP con un redirect URL, mediante lo user-agent dell'utente.

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

code=usDwMnEzJPpG5oaV8x3j &state=fyZiOL9Lf2CeKuNT2JzxiLRDink0uPcd
```

Esempio 18: Esempio di Response di Authorization endpoint

4.3 Token Request

Per ottenere un token di accesso, tramite il quale poter effettuare la richiesta di attributi, il SP effettua una richiesta all'endpoint */token* dell'AS inviando l'**authorization code** ottenuto con la precedente authorization request, all'interno nel parametro *code* e i restanti parametri come definiti in *private_key_jwt* Client Authentication di OIDC.

```
POST /token HTTP/1.1
Host: aa.it/
Content-Type: application/x-www-form-urlencoded

client_id=https%3A%2F%2Frp.spid.agid.gov.it&
client_assertion=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IlNQSUQiLCJhZG
1pbiI6dHJ1ZX0.LVyRDPVJm0S9q7oiXcYVIIqGWY0wWQlqxvFGYswLF88&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer&
code=usDwMnEzJPpG5oaV8x3j&
code_verifier=9g8S40MozM3NSqjHnhi7OnsE38jklFv2&
grant_type=authorization_code
```

Esempio 19: Token Request

La Token Request deve utilizzare il metodo di autenticazione *private_key_jwt*[14] e contenere i seguenti parametri:

Parametro	Descrizione	Valori ammessi
client_id	URI che identifica univocamente il SP come da Registro SPID.	*Deve corrispondere all'EntityID del SP*

[14] private_key_jwt – OpenID Connect Core, 9. Client Authentication (https://openid.net/specs/openid-connect-core-1_0.html#ClientAuthentication)

client_assertion	JWT firmato con la chiave privata del SP contenente intestazioni e payload comuni relative al SP	
client_assertion_type		**urn:ietf:params:oauth:client - assertion-type:jwt-bearer**
code	*authorization code* restituito con l'*Authorization Response*	
code_verifier	Codice di verifica del code_challenge inviato	
grant_type	Tipo di credenziale presentata dal SP per la richiesta corrente.	Può assumere uno tra i seguenti valori: **authorization_code refresh_token**
refresh_token	Da utilizzare per richieste continuative	Solo se grant_type è **refresh_token**

Table 11: Parametri Token Request

4.4 Token Response

Alla ricezione di una Token Request, l'AS, verifica l'autenticità della richiesta controllando la firma della *client_assertion* e la corrispondenza del **code_verifier** per *PKCE*[15]. Assicurata l'autenticità della richiesta restituisce una response che include l'**access token** ed eventualmente, nel caso in cui sia stato specificato nel parametro scope della authorization request il valore offline_access, un **refresh token**. Access token e refresh token devono essere JWT firmati e cifrati dall'AS il cui payload deve contenere almeno le seguenti informazioni.

Nell'header, i claim:

Parametro	Descrizione
typ	valorizzato con la stringa "JOSE" ad indicare che il token è un JWT (JWS o JWE) secondo le specifiche JWS Compact Serialization o JWE Compact Serialization.
alg	valorizzato con l'identificativo JWA dell'algoritmo crittografico utilizzato per l'autenticazione JWE.
x5c	valorizzato con il certificato o la catena dei certificati, in formato X.509, corrispondente alla chiave pubblica del certificato di sigillo elettronico utilizzato. Il certificato o la catena dei certificati sono codificati come array JSON di stringhe corrispondenti ai valori dei certificati in formato DER. Ogni stringa nell'array è codificata in Base64. Il certificato contenente la chiave pubblica del sigillo utilizzato per firmare il token deve essere la prima stringa dell'array.
x5t#S256	valorizzato con l'hash del certificato corrispondente alla chiave pubblica del certificato di sigillo elettronico utilizzato. Il destinatario del token deve poter recuperare il certificato e validarlo.

Table 12: Claim all'interno della intestazione JWT

Nel payload in formato JWT dovranno risultare essere presenti almeno i seguenti claim:

Parametro	Descrizione
jti	Identificatore unico del token che è possibile utilizzare per prevenirne il riuso illecito. Deve essere di difficile individuazione da parte di un attaccante e composto da una stringa casuale

[15] Proof Key for Code Exchange by OAuth Public Clients (https://tools.ietf.org/html/rfc7636)

iss	Identificatore del mittente del messaggio (SP o AA) che lo contraddistingue univocamente nella federazione.
aud	Identificatore del destinatario del messaggio (AA o SP) che lo contraddistingue univocamente nella federazione
iat	Data/ora di emissione del token codificato come UNIX timestamp
exp	Data/ora di scadenza del token codificato come Unix timestamp.
sub	identificativo univoco del soggetto autenticato

Table 13: Claim all'interno del payload JWT

Nel payload in formato JWT, opzionalmente i claim:

Parametro	Descrizione
acr	Livello di autenticazione SPID effettivo con il quale l'utente ha eseguito l'autenticazione.
scope	se *offline_access*, indica che l'autorizzazione è valida per una richiesta continuativa.

Table 14: Parametri Authorization Response

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

{
 "access_token": " … ",
 "token_type": "Bearer",
 "refresh_token": " … ",
 "expires_in": 3600,
 "id_token": " … "
}
```

Esempio 20: Esempio di token response

4.5 Introspection Request

L'AS espone l'endpoint /introspection che permette ad una AA standalone di verificare la validità di access token e refresh token. La Introspection Request deve utilizzare il metodo di autenticazione *private_key_jwt*[16] e contenere i seguenti parametri:

Parametro	Descrizione	Valori ammessi
client_id	URI che identifica univocamente il SP come da Registro SPID.	*Deve corrispondere all'EntityID del SP*

[16] private_key_jwt – OpenID Connect Core, 9. Client Authentication (https://openid.net/specs/openid-connect-core-1_0.html#ClientAuthentication)

client_assertion	JWT firmato con la chiave privata del SP contenente intestazioni e payload comuni relative al SP	
client_assertion_type		**urn:ietf:params:oauth:client - assertion-type:jwt-bearer**
token	Il token per il quale si vogliono ottenere informazioni	

Table 15: Parametri Introspection Request

Esempio di Token Introspection Request:

```
POST /introspection HTTP/1.1
Host: aa.it
Content-Type: application/x-www-form-urlencoded

client_assertion=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IlNQSUQiLCJhZG
1pbiI6dHJ1ZX0.LVyRDPVJm0S9q7oiXcYVIIqGWY0wWQlqxvFGYswLF88&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer&
client_id=https%3A%2F%2Frp.spid.agid.gov.it&
token=$jwt_access_token
```

Esempio 21: Token introspection endpoint, esempio di richiesta

4.6 Introspection Response

La Token Response contiene i seguenti parametri:

Parametro	Descrizione	Valori ammessi
client_id	URI che identifica univocamente il SP al quale è stato rilasciato il token	*Deve corrispondere all'EntityID del SP*
active	Valore booleano che indica la validità del token. Se il token è scaduto, è revocato o non è mai stato emesso per il client_id chiamante, l'Introspection Endpoint deve restituire false.	
scope	Gli scopi per i quali l'access_token è stato rilasciato.	
exp	Scadenza del token.	
sub	identificativo univoco del soggetto autenticato	

Table 16: Parametri Introspection Response

Esempio di Token Introspection Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
```

```
{
  "active": true,
  "scope": "openid get-qualifica",
  "exp": 1519033149,
  "client_id": "https://rp.agid.gov.it/",
  "sub": "64d799c8-969f-430e-bbc5-fab21c606bd8"
}
```


Esempio 22: Token introspection endpoint, esempio di response

5. Autorizzazione per richieste continuative

Nel caso di richieste continuative, nella Authorization Request dovrà essere specificato nel parametro **scope** il valore **offline_access**. Una Authorization Request di tipo continuativo permette di avere nella Token Response, oltre all'access token, anche un refresh token. Il refresh token potrà essere utilizzato presso l'endpoint */token* specificando il valore refresh_token per il parametro grant_type, al fine di ottenere un nuovo access token con cui continuare ad effettuare richieste di attributi, senza dover ripetere l'intero processo di autenticazione.
Il refresh token può essere utilizzato fino alla scadenza del periodo concordato all'atto dell'autenticazione oppure fino a quando non viene esplicitamente revocato dall'utente tramite il **Servizio di consultazione per l'utente.** Il servizio di consultazione per l'utente è una implementazione applicativa, interfaccia, esposta dalla AA che consente all'utente di prendere visione dei consensi rilasciati che lo riguardano. L'utente detiene il diritto di revocarli in qualsiasi momento. Un SP o una AA per conoscerne la validità di un consenso, può controllare lo stato di un token (access o refresh token) presso l'*introspection endpoint* dell'AS.

Indice delle tabelle e delle illustrazioni

Bibliografia

Fielding, Roy Thomas (2000). "Representational State Transfer (REST)". *Architectural Styles and the Design of Network-based Software Architectures* (PhD). University of California, Irvine
Modello di interoperabilità per la Pubblica Amministrazione. (https://www.agid.gov.it/it/infrastrutture/sistema-pubblico-connettivita/il-nuovo-modello-interoperabilita)
Linee Guida SPID OIDC. (https://docs.italia.it/AgID/documenti-in-consultazione/lg-openidconnect-spid-docs/it/bozza/index.html)
AARC-G049 - A specification for IdP hinting. (https://aarc-project.eu/wp-content/uploads/2019/04/AARC-G049-A_specification_for_IdP_hinting-v6.pdf)
Open API Specification v3. (https://swagger.io/specification)
Bearer Token Authentication. (https://tools.ietf.org/html/rfc6750)
JSON Web Token (JWT). (https://tools.ietf.org/html/rfc7519)
RFC8725 - JSON Web Token Best Current Practices (https://tools.ietf.org/html/rfc8725)
Semantic Version Specification. (https://semver.org/)

RFC7807 - Problem Details for HTTP APIs (https://tools.ietf.org/rfc/rfc7807.txt)
The OAuth 2.0 Authorization Framework (https://tools.ietf.org/rfc/rfc6749.txt)
private_key_jwt – OpenID Connect Core, 9. Client Authentication
(https://openid.net/specs/openid-connect-core-1_0.html#ClientAuthentication)