Elements de repères OpenID Connect (vs CAS & shibboleth)

Authorization code flow

C'est le plus classique, proche de CAS et/ou proxy CAS :

CAS	OIDC	SAML 2
Server	OP (OpenID Provider) ou Authorization Server	IDP
Service	Client ou RP (Relying Party)	SP
/login	/authorization	
/serviceValidate /p3/serviceValidate	/token	HTTP Artifact
	/userinfo	Attribute Query
Paramètres :		
`service`	`client_id` & `redirect_uri` & `state`	`SAMLRequest`
`ticket`	`code`	Artifact
`gateway`	`prompt=none`	`isPassive`
`renew`	`prompt=login`	`ForceAuthn`
Un peu similaire :		
proxy ticket (valide une fois)	access token (valide un certain temps)	
PGT	refresh token	

Comparé à CAS, le client (service) doit s'enregistrer sur l'IDP pour avoir un "`client_id`" et un "`client_secret`".

Autres code flow

Flow	response_type	response_mode
Authorization code	"`code`"	`query`
Implicit	"`id_token token`" ou "`id_token`"	`fragment`
Hybrid	"`code id_token`" ou "`code token`" ou "`code id_token token`"	`fragment`

Dans le cas "`response_type=id_token`", l'`id_token` contient toutes les claims demandés par le paramètre "`scope`".

Dans les autres cas, les claims doivent être récupérés en faisant une requête `/userinfo` avec l'access token.

response_mode

- `query` (CAS, SAML "HTTP Redirect")
- `fragment` (implicit grant, #token=xxx, était possible avec CAS < 3.4.4 en mettant un "#" dans l'url "service", mais ne donnait accès qu'au ticket, pas plus)
- `form_post` (SAML "HTTP POST")

expérimental :

- CORS AJAX
- `postmessage` (notamment chez google en non standard avec `redirect_uri=postmessage` : http://www.riskcompletefailure.com/2013/03/postmessage-oauth-20.html)

http://connect2id.com/products/server/docs/config/core#op-authz-responseModes
https://openid.net/specs/oauth-v2-multiple-response-types-1_0.html

Mapping eduPerson attributes to OIDC claims

https://wiki.refeds.org/display/GROUPS/Mapping+SAML+attributes+to+OIDC+Claims