Implementació d'un proveïdor de servei PAPI (PoA) lleuger en Python

Presentació del projecte final de màster

Javier Peña Blázquez
jpenyab@uoc.edu

UNIVERSITAT OBERTA DE CATALUNYA

http://www.uoc.edu

26 de Maig de 2013

Contingut

Introducció

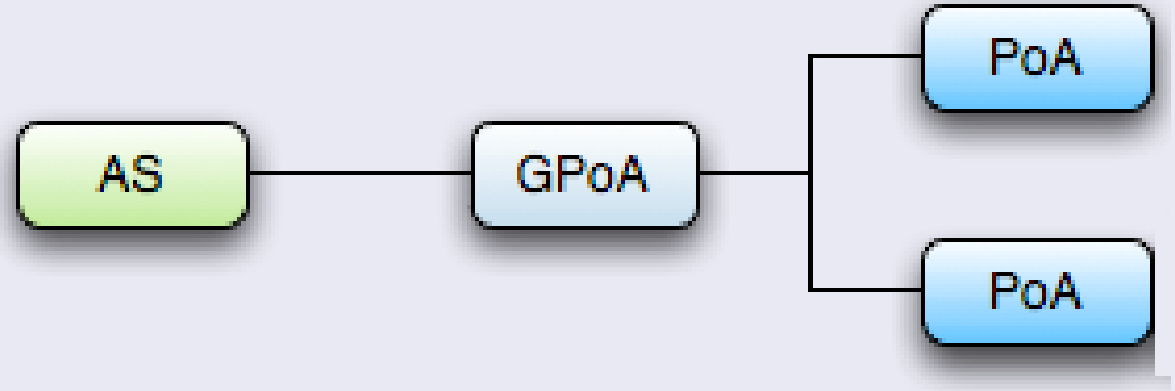
Proveïdor d'Accés a Punts d'Informació (PAPI)

- Protocol obert desenvolupat per RedIRIS.
- Es fa servir per implantar sistemes SSO (**Single Sign-On**)

Origen i evolució de PAPI

- Surt per la necessitat que tenen les **biblioteques virtuals** d'oferir accés als recursos digitals per als usuaris.
- Solució: RedIRIS proposa un model que separa l'**autenticació** del **control de accés** → PAPI.
- PAPI evoluciona i es realitzen implementacions en diversos llenguatges de programació. És quan es comença a parlar de PAPI com a **protocol**.

Arquitectura de PAPI



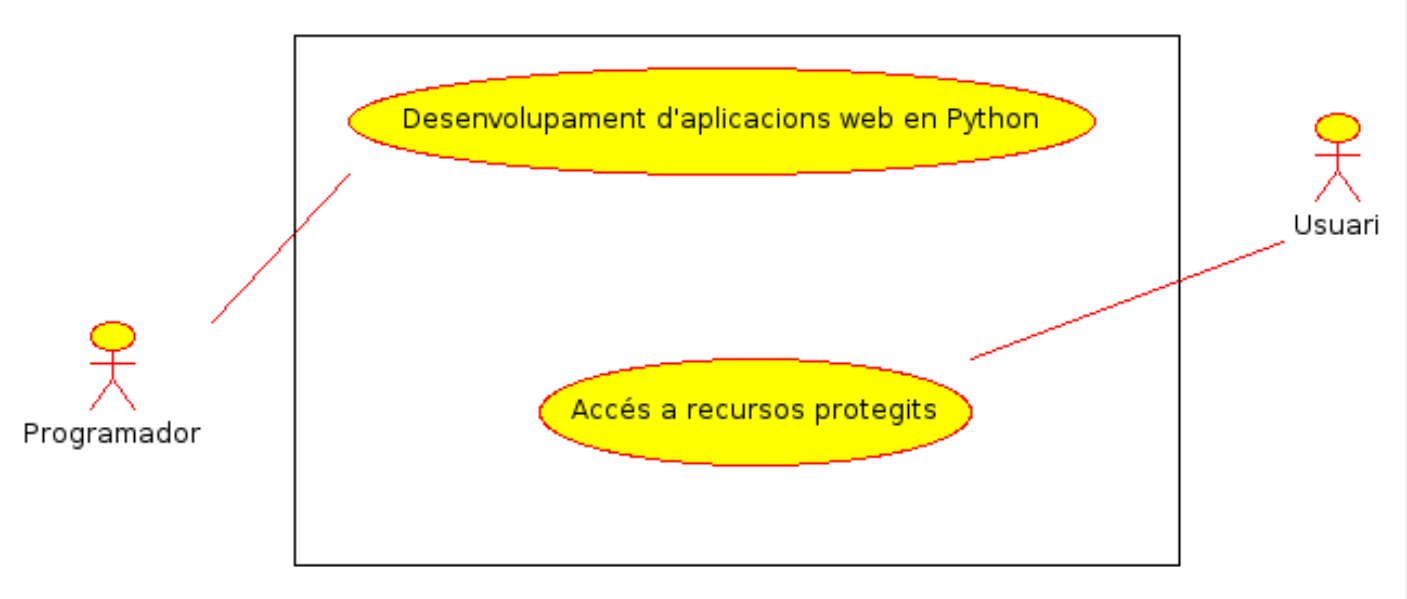
Estat de la tècnica

- Implementacions estables dels components PAPI en: Perl, Java, PHP, C i .Net.
- No es disposa de cap component implementat en **Python**.

Objectius del projecte

- **Analitzar el protocol PAPI.**
- **Dissenyar i programar una biblioteca en Python que implementi un PoA de PAPI.**

Abast del sistema



- Aspectes econòmics, tècnics i legals

Estudi de la situació actual

- La llibreria **EasyPyPoA** és genèrica i no està destinada a ser implantada en una sola empresa.

Estudi de viabilitat

Definició dels requisits

- Funcionalitat propia d'un agent PoA de PAPI.
- Desenvolupada en Python.
- Funcional en qualsevol servidor web amb suport per Python.
- Complir amb la "Style Guide for Python Code"
- Interficie ben definida.
- Documentada i publicada sota una llicència lliure.
- Despeses el més baixes posibles.

Estudi, valoració i selecció de les alternatives de solució

- Una única alternativa de solució → Desenvolupar una llibreria (**EasyPyPoA**).
- Aquesta alternativa permet complir amb els requisits definits.

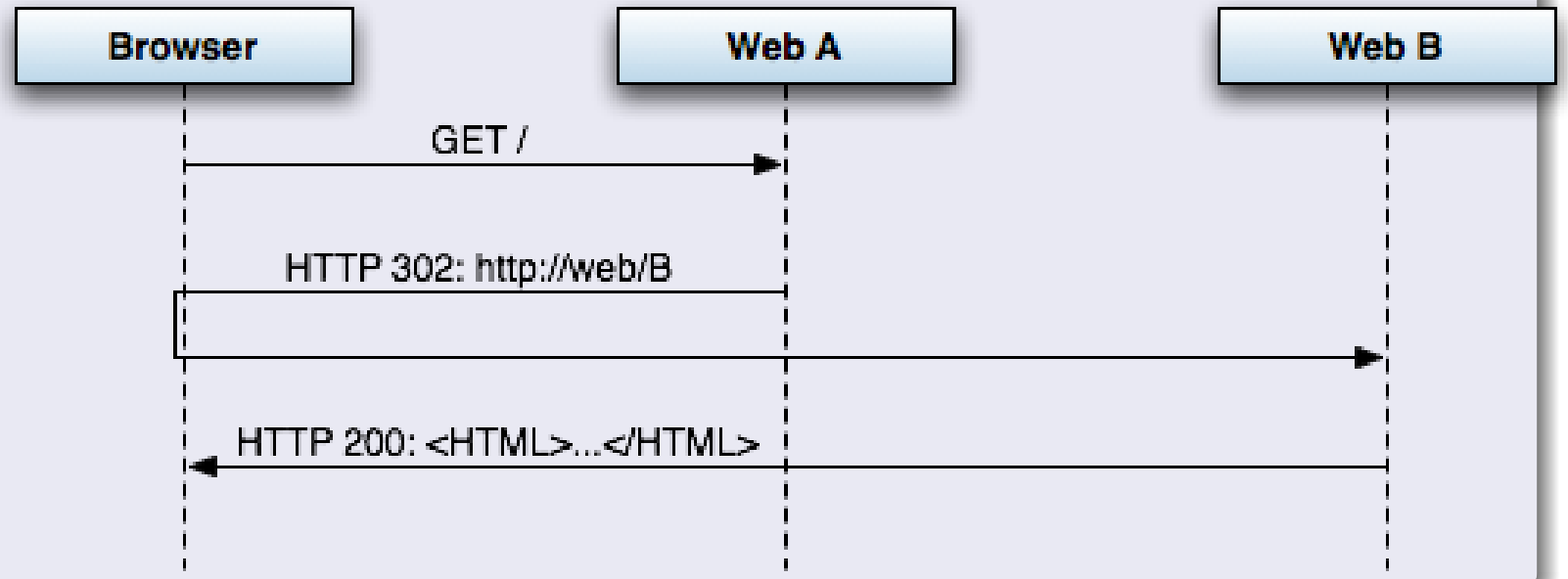
Definició del sistema

La llibreria ha de ser **lleugera**:

- No ha d'implementar la comunicació directa entre PoA i AS.
- El PoA no ha de realitzar filtres d'accés.
- Les dades d'autenticació s'han de mantenir en una cookie de sessió.

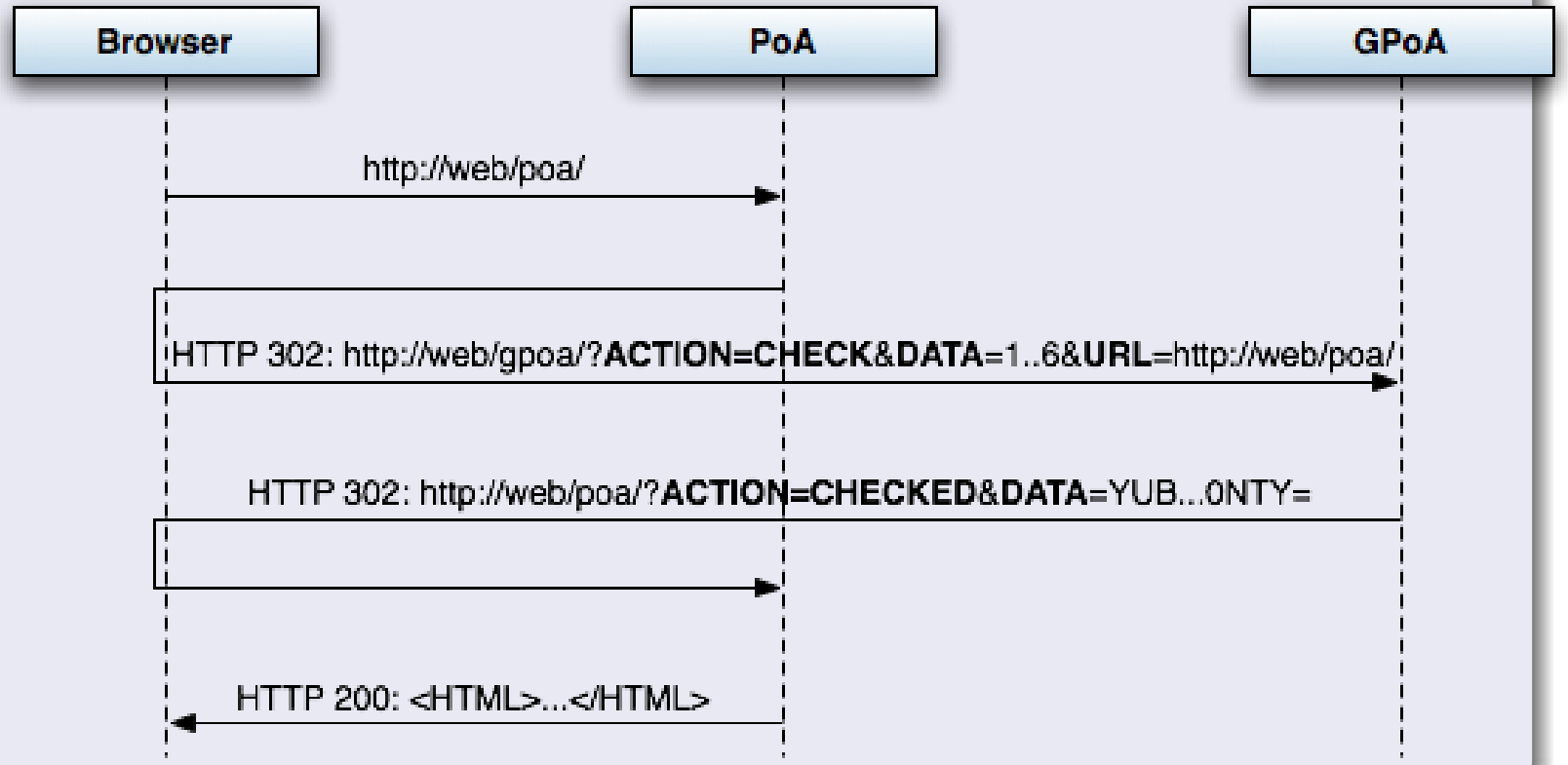
Establiment de requisits

- **Requisit "Comunicació entre el PoA i el GPoA"**



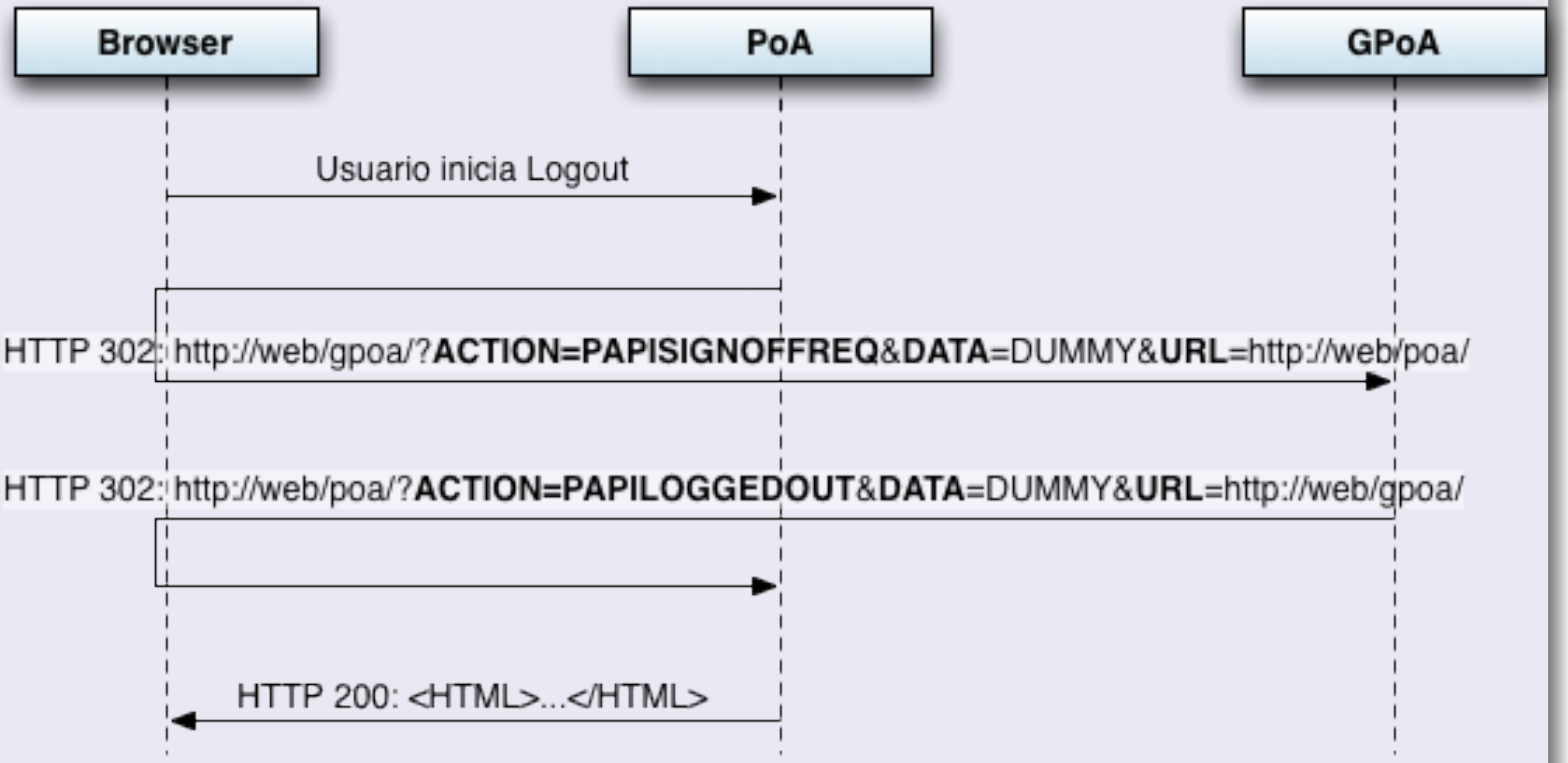
Establiment de requisits

- **Requisit "Accés a un recurs protegit i petició d'autorització: CHECK - CHECKED"**



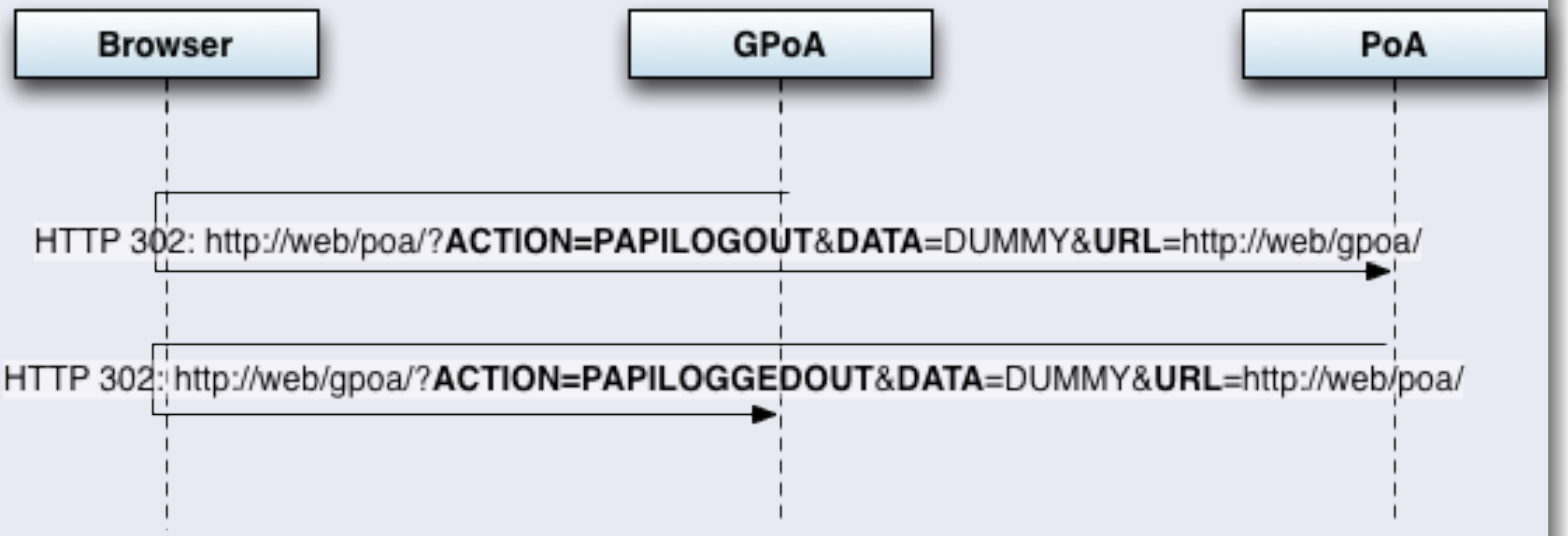
Establiment de requisits

- **Requisit "Procés de Logout"**



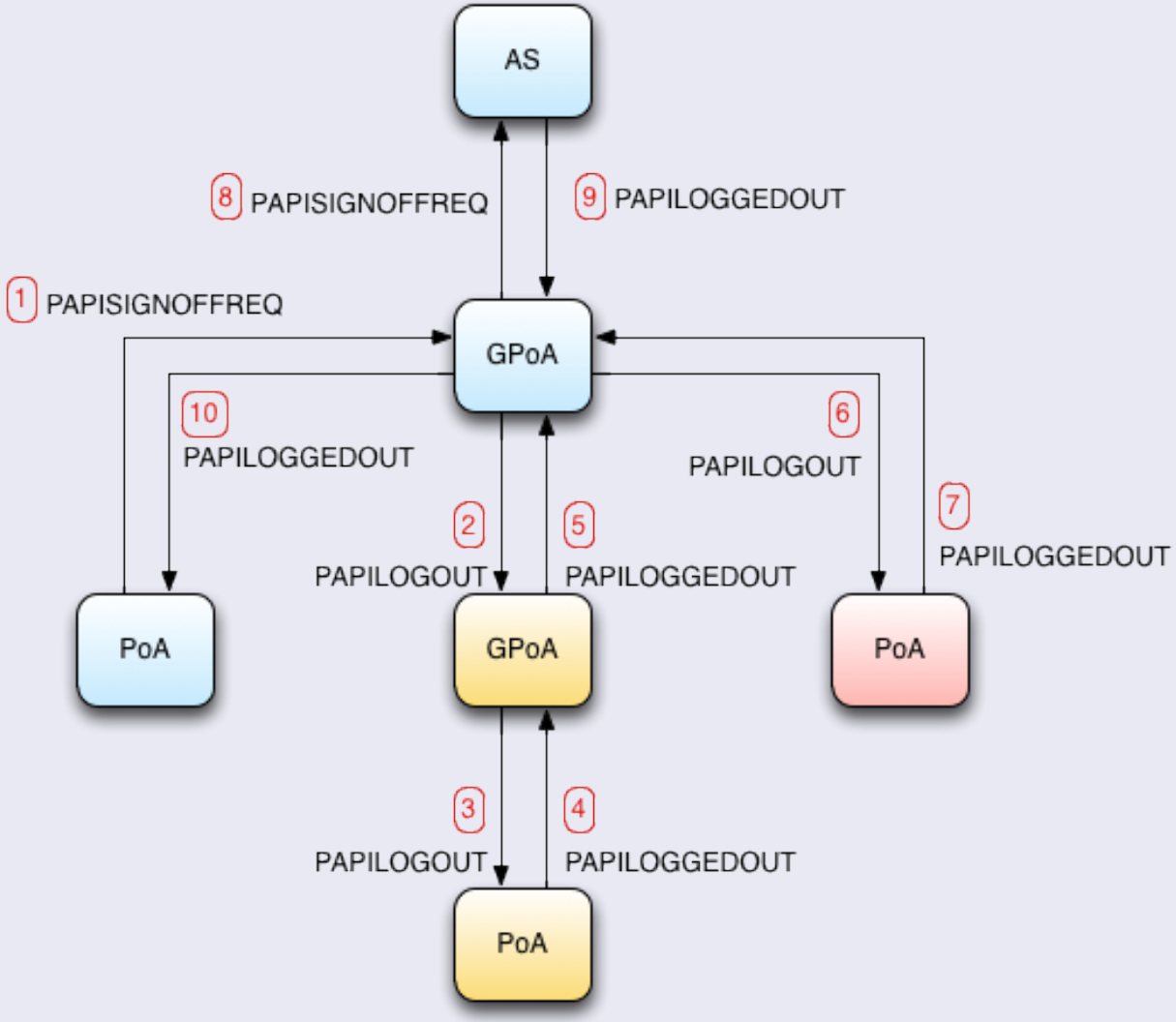
Establiment de requisits

- **Requisit "Procés de Logout"**



Establiment de requisits

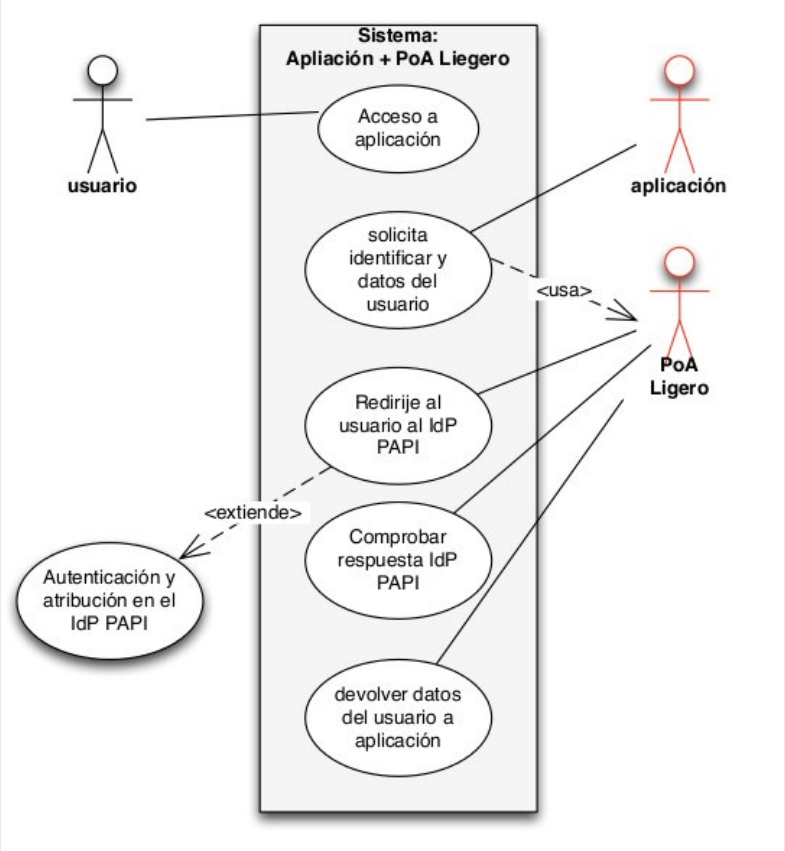
- **Escenari complet de Logout**



Casos d'ús

- **Cas d'ús "Protecció i accés a un recurs protegit"**

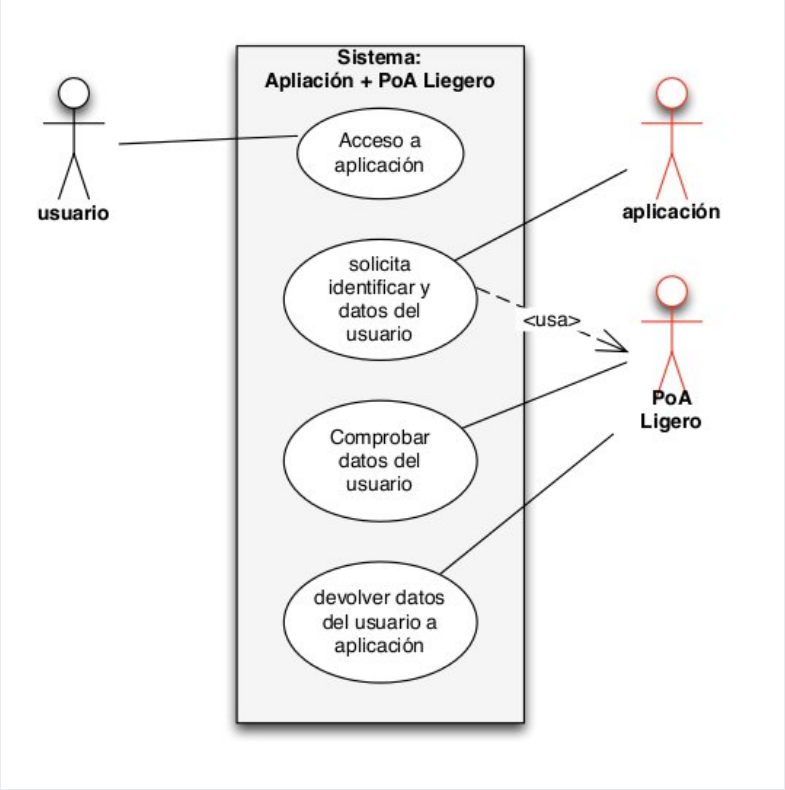
Context: Un usuari accedeix per primera vegada a una aplicació protegida per un PoA lleuger PAPI



Casos d'ús

- **Cas d'ús "Protecció i accés a un recurs protegit"**

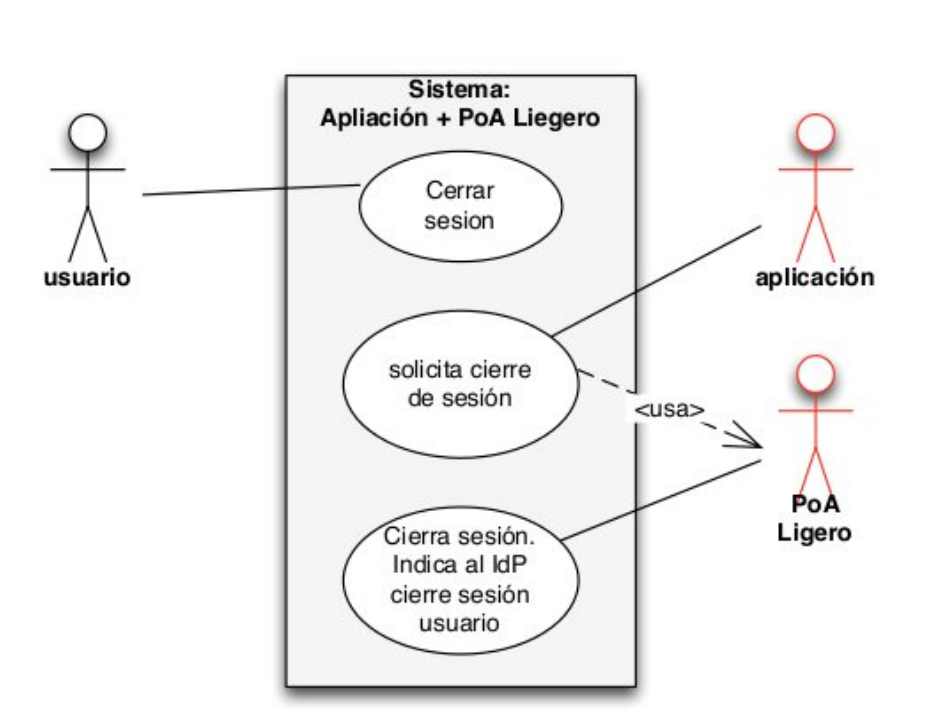
Context: Un usuari ja autenticat accedeix de nou a una aplicació protegida per un PoA lleuger PAPI.



Casos d'ús

- **Cas d'ús "Procés de logout en PAPI"**

Context: Un usuari desitja sortir d'una aplicació protegida per un PoA lleuger PAPI, en la qual té una sessió iniciada.



Casos d'ús

- **Cas d'ús "Procés de logout en PAPI"**

Context: L'usuari tanca la sessió en una altra aplicació integrada en el SSO, la qual cosa provoca que el GPoA envii un missatge de Logout al PoA.

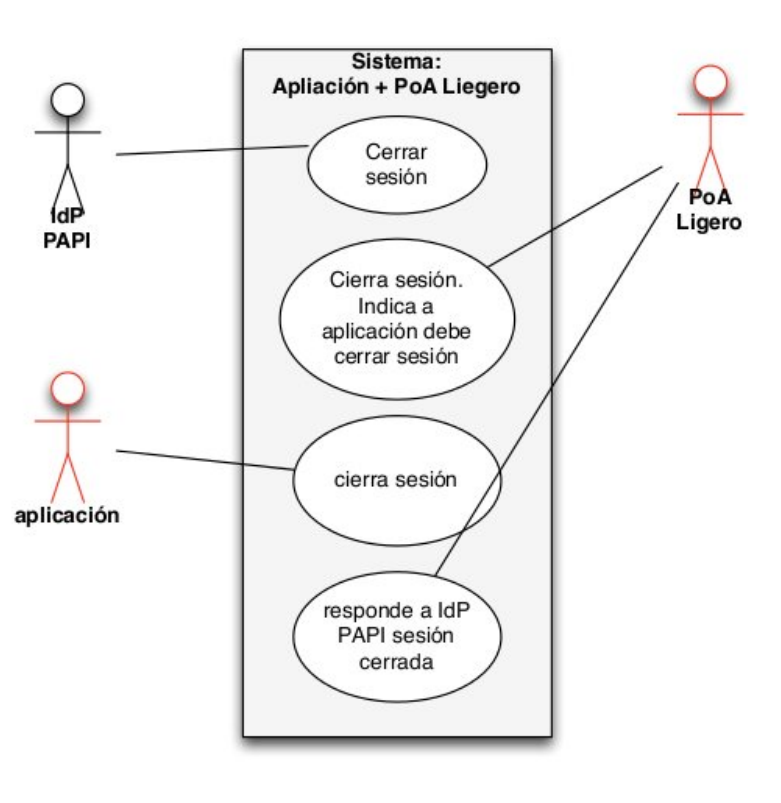


Diagrama de components

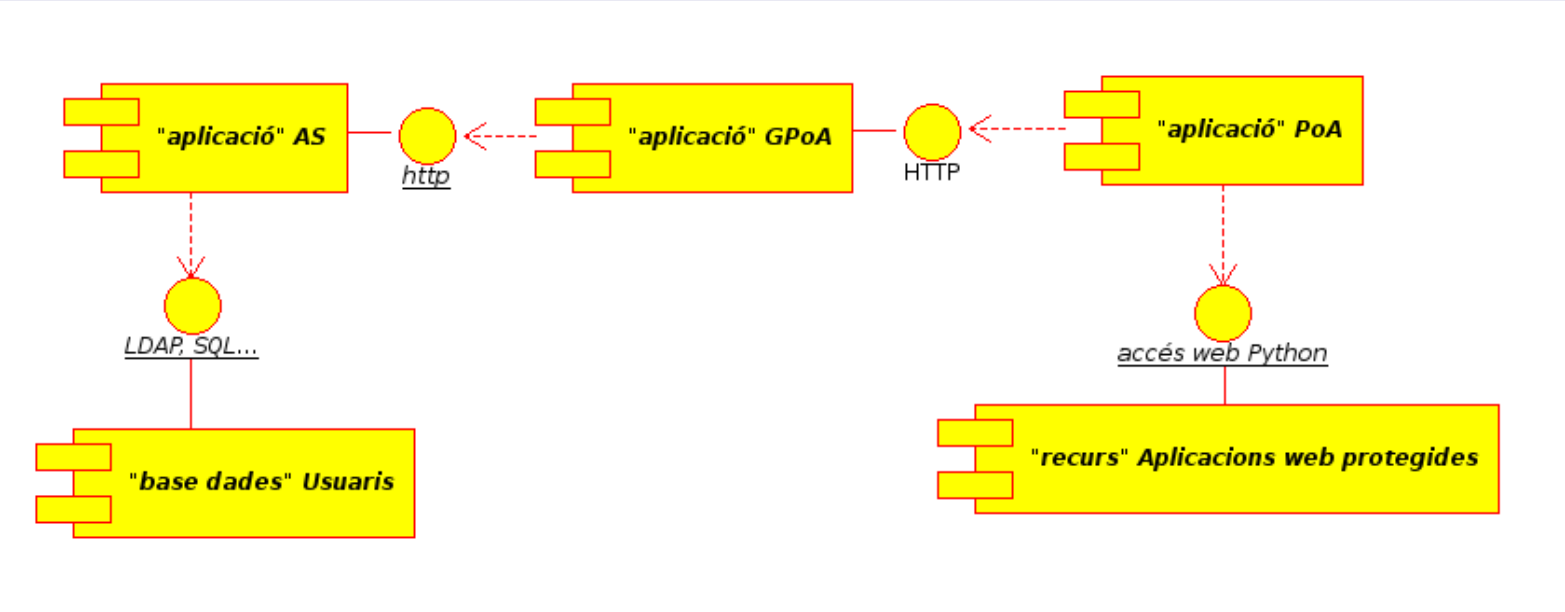
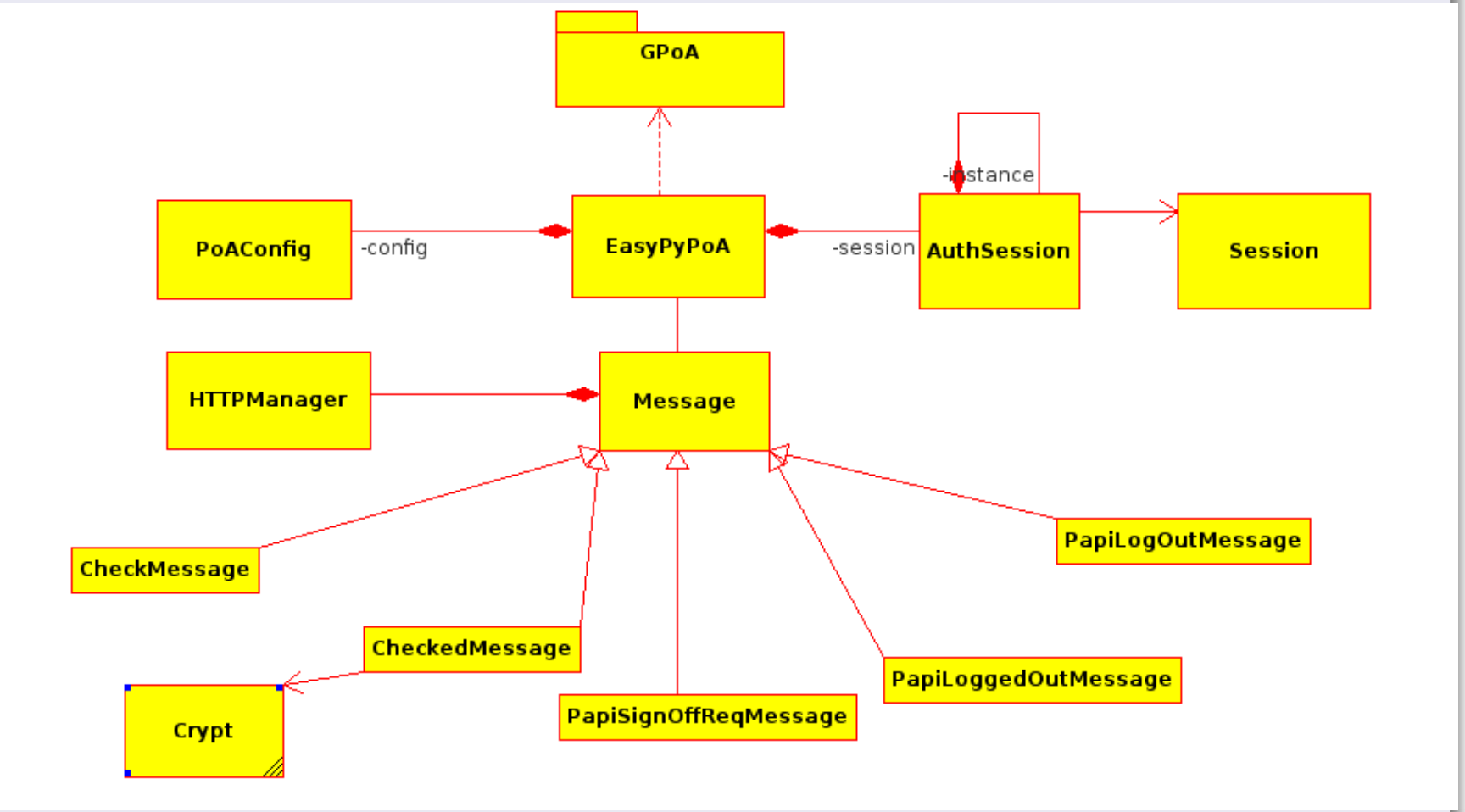


Diagrama de classes



Desenvolupament, implantació i manteniment

Desenvolupament

- Concretar versions de programari i llibreries.
- Estudiar les llibreries.
- Implantar l'IDE.
- Desenvolupar les proves unitàries.
- Desenvolupar les classes.
- Realitzar la documentació
- Desenvolupar les proves d'integració.
- Aprovar el sistema

Implantació

Tasques:

- Instal·lació del programari necessari (servidor web Apache, python, openssl i m2crypto).
- Instal·lació de la llibreria EasyPyPoA.
- Configuració del PoA.

Protecció d'una aplicació web Python:

```
...
try:
    poa = EasyPyPoA('TestEasyPoA')
    if poa.check_access():
        body = '<h1>User has access permissions</h1>'
        userData = poa.get_user_data()
...
```

Desenvolupament

- Intent d'accés a una aplicació web protegida per part d'un usuari que no ha iniciat ninguna sessió al sistema PAPI.

Name / Path	Method	Status / Text	Type	Size / Content	Timeline
index.py /EasyPyPo	GET	302 Found	text/html	938 B / 0 B	
gpoa1.py /EasyPyPo	GET	302 Found	text/html	946 B / 0 B	
index.py3 /EasyPyPo	GET	200 OK	text/html	516 B / 376 B	

- Intent d'accés a una aplicació web protegida per part d'un usuari que sí ha iniciat una sessió previament al sistema PAPI

Name / Path	Method	Status / Text	Type	Size / Content	Timeline
index.py /EasyPyPo	GET	200 OK	text/html	516 B / 376 B	

Conclusions

Objectius aconseguits

- Comprensió elavada sobre el funcionament de PAPI.
- Desenvolupament d'una llibreria que implementa un PoA PAPI lleuger.

Dificultats trobades

- Comprensió del que havie de ser una llibreria "lleugera".
- Implementació de sessions en Python.
- Xifrat PAPI en Python → M2Crypto

Conclusions

Possibilitats d'ampliació

- Compatibilitat de EasyPyPoA amb Python 3.
- Implementació d'un GPoA en Python.

Conclusions

- Contribució a un projecte de programari lliure → PAPI.
- Repte personal per a l'alumne.