Contribution ID: **47** Type: **not specified**

# Scalable management with Seafile SPORE specification

*Wednesday, 1 February 2017 14:20 (20 minutes)*

The Strasbourg University provides a file synchronization and sharing service via Seafile software. This service is delivered to more than 11000 researchers, teachers and employees which are distributed across more than 900 structures, such as schools, laboratories and departments. At this scale, a centralized management of the accounts, guest accounts, groups, shared repositories and space quotas is not an option.

The idea of this paper is to propose a simple and generic mechanism to distribute and delegate this management tasks to each structure or user requesting a custom configuration.

Providing such a scalable and distributed management solution is an interesting problem, because it must:

- at one side, request the University structured data repository in order to compute the rights relative to a given user;
- at the other side, be able to communicate with Seafile to inject the computed custom parameters.

To this end, we decided to create the Seafile SPORE description file (cf. [seaa]) based on the well documented Seafile REST API (cf. [seab]).

A SPORE description (cf. [spo]) is a simple JSON document, which describes a service HTTP API, in order to dynamically or statically generate high level client objects. Lots of SPORE implementations are available, such as: Clojure, Javascript, Lua, Nodejs, Perl, Python, Ruby, Livescript and Groovy.

Moreover SPORE description is the main format used to define, manage and interconnect Strasbourg University software.

Therefore, the first section of this paper presents the Seafile SPORE specification publicly available (cf. [seaa]).

Then, the second section details the guest accounts management, which includes:

1. The interconnection process between the University LDAP and the Seafile service: it is implemented in Python on top of the Britney Python SPORE client (cf. [bri]).
2. The dedicated web pages: this part is developed as a “uWSGI” service with “bottle” a Python web framework (cf. [bot]) and integrated to the HTTP “nginx” server which is used for the Seafile frontend.
3. The integration inside Seafile web interface: the link to create guest accounts is added to Seafile web interface and the authentication is done directly by Seafile Django service.

This functionality is heavily used for building external research collaboration and to create student accounts.

Furthermore and based on the same mechanisms, the third section depicts on going feature implementations such as group, shared repositories and quotas management.

Finally, we conclude with the benefits and limitations of this approach and reviews which load of management tasks can be delegated.

### References:

[bot] Bottle: Python web framework.
http://bottlepy.org

[bri] Python implementation of spore (based on spyre).
https://github.com/agrausem/britney

[seaa] Seafile spore description file.
http://rest-api.u-strasbg.fr/seafile/description.json

[seab] Synchronization algorithm | seafile server manual.
https://manual.seafile.com/develop/web_api_v2.1.html

[spo] Spore - specification to a portable rest environment.
https://spore.github.io/

**Primary authors:** Mr FINKBEINER, Sébastien (Strasbourg University); LUCAS, Vincent (Strasbourg University)

**Presenters:** Mr FINKBEINER, Sébastien (Strasbourg University); LUCAS, Vincent (Strasbourg University)

**Session Classification:** New site services