Nooda

Une plateforme de communication spontanée

Fabrice Ben Hamouda et Ludovic Patey

`https://nooda.net`

Gestion des données sur le Web

Professeurs : Serge Abiteboul et Pierre Senellart

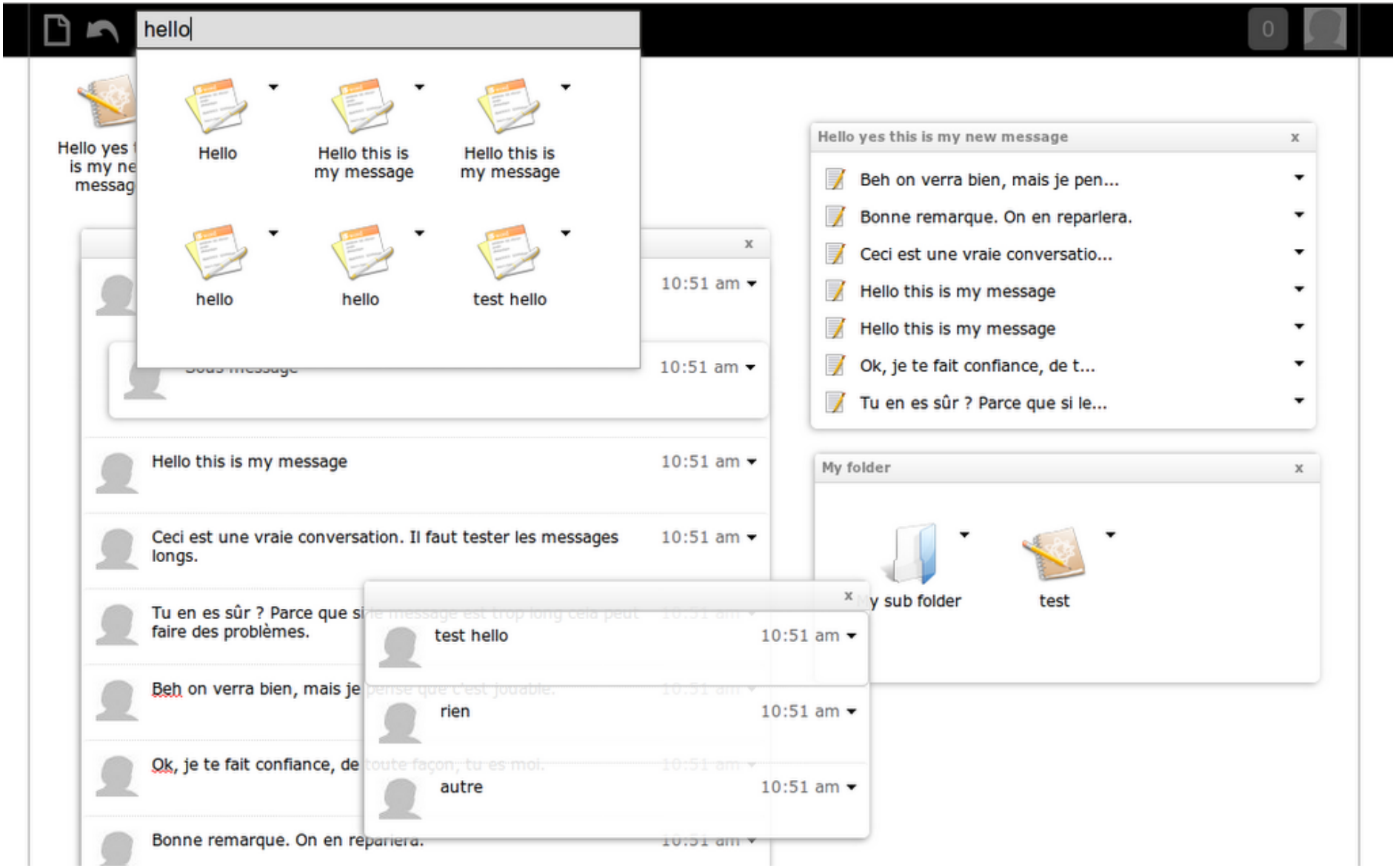
24 Novembre 2011

Sommaire

Plan

Démonstration

`http://tinyurl.com/d8t559r`


hello
Hello
Hello this is my message
Hello this is my message
hello
hello
test hello
Hello yes this is my new message
Beh on verra bien, mais je pen...
Bonne remarque. On en reparlera.
Ceci est une vraie conversatio...
Hello this is my message
Hello this is my message
Ok, je te fait confiance, de t...
Tu en es sûr ? Parce que si le...
My folder
y sub folder
test
10:51 am
Hello this is my message
Ceci est une vraie conversation. Il faut tester les messages longs.
Tu en es sûr ? Parce que si
faire des problèmes.
Beh on verra bien, mais je
Ok, je te fait confiance, de
Bonne remarque. On en reparlera.
test hello
rien
autre

Plan

Démonstration

Gestion des conflits

nooda_page_aaaa

nooda_page_temp

Nooda est une plateforme

nooda_page_aaaa

nooda_page_temp

Nooda est une plateforme

$\longrightarrow$ réplication $\longrightarrow$

Nooda est une plateforme

Nooda est une plateforme

nooda_page_aaaa

Nooda est une plateforme

nooda_page_temp

$\longrightarrow$ réplication $\longrightarrow$

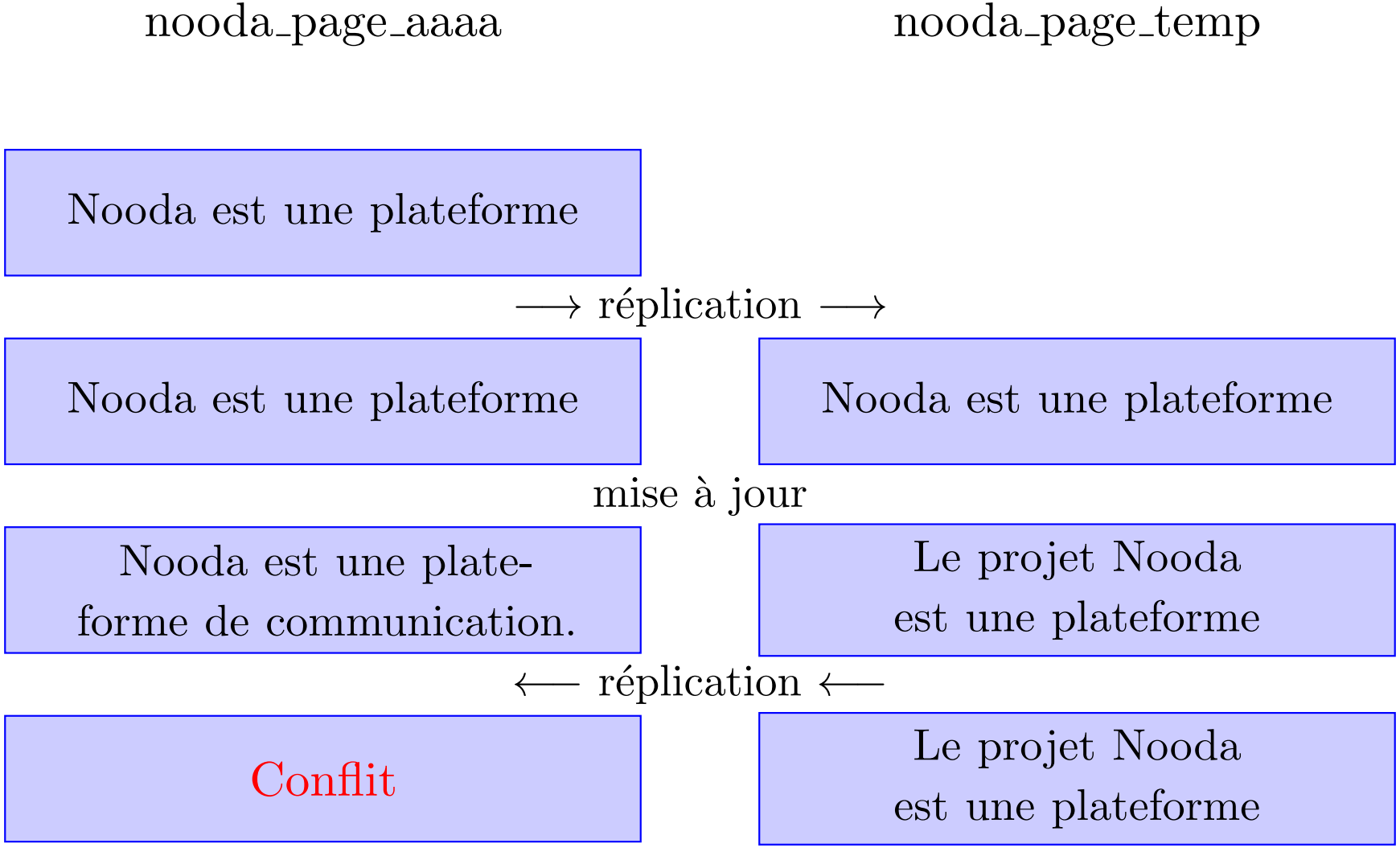
Nooda est une plateforme

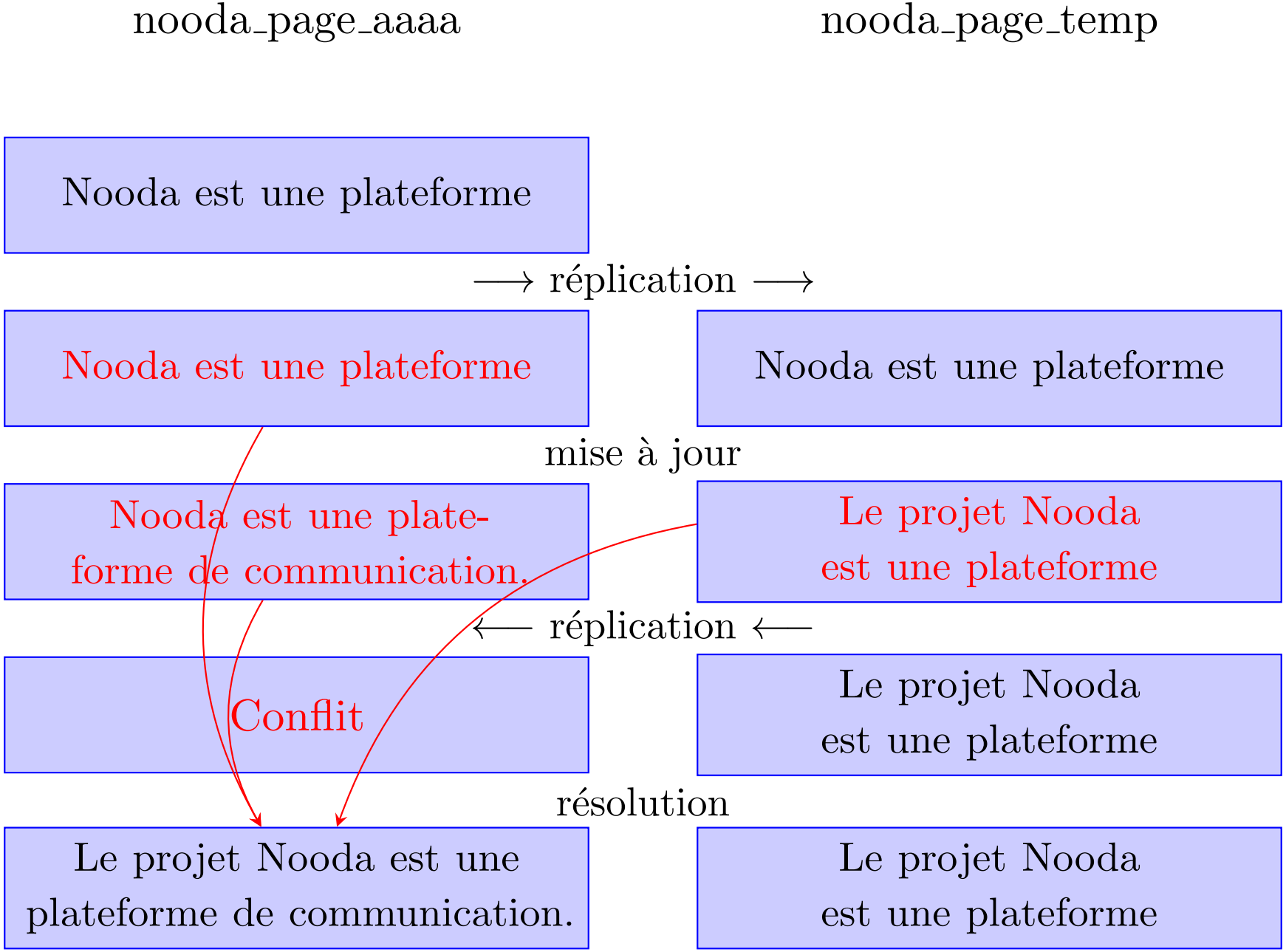
Nooda est une plateforme

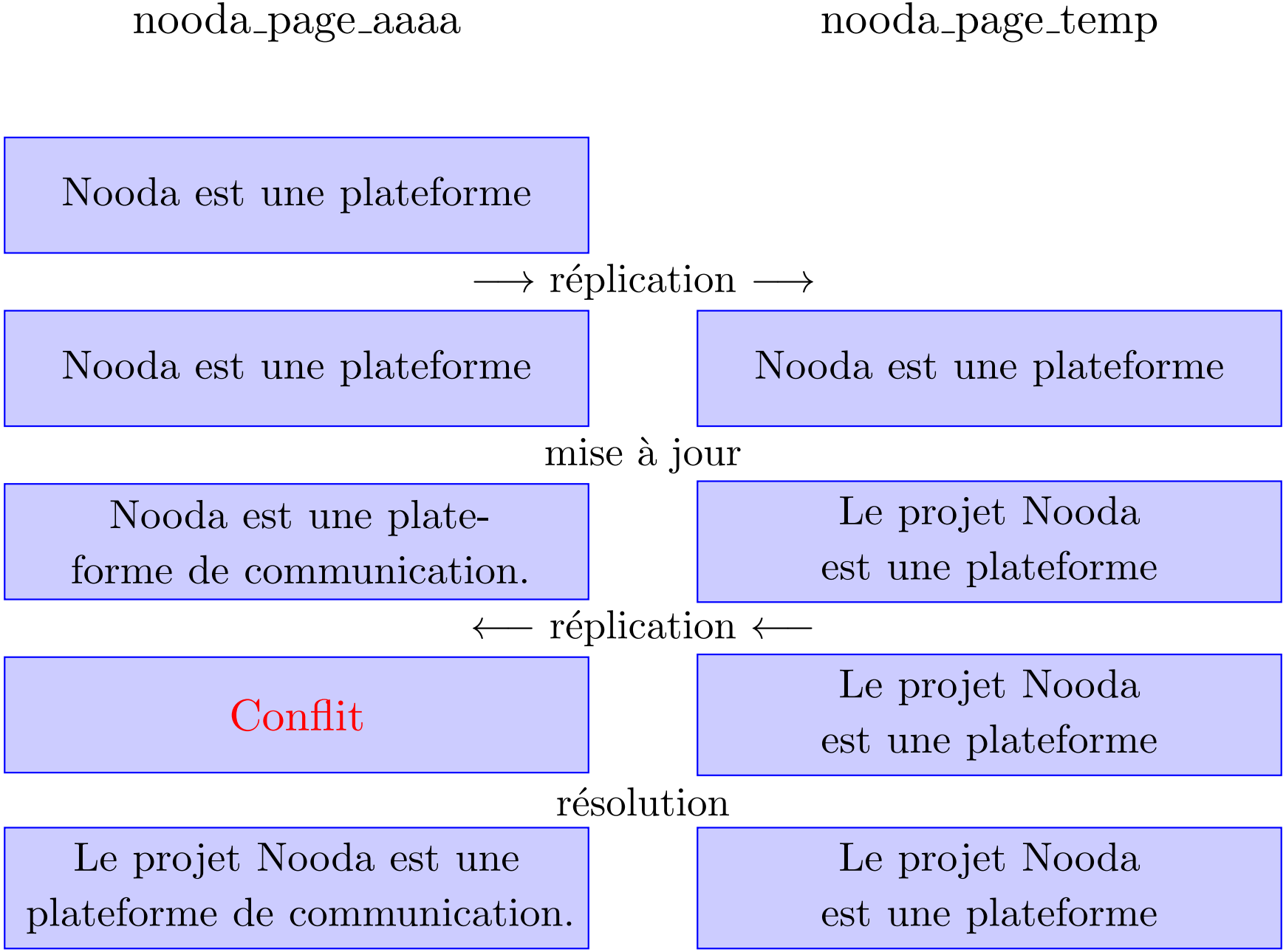
mise à jour

Nooda est une plate-
forme de communication.

Le projet Nooda
est une plateforme


nooda_page_aaaa
nooda_page_temp
Nooda est une plateforme
⟶ réplication ⟶
Nooda est une plateforme
Nooda est une plateforme
mise à jour
Nooda est une plate-
forme de communication.
Le projet Nooda
est une plateforme
⟵ réplication ⟵
Conflit
Le projet Nooda
est une plateforme

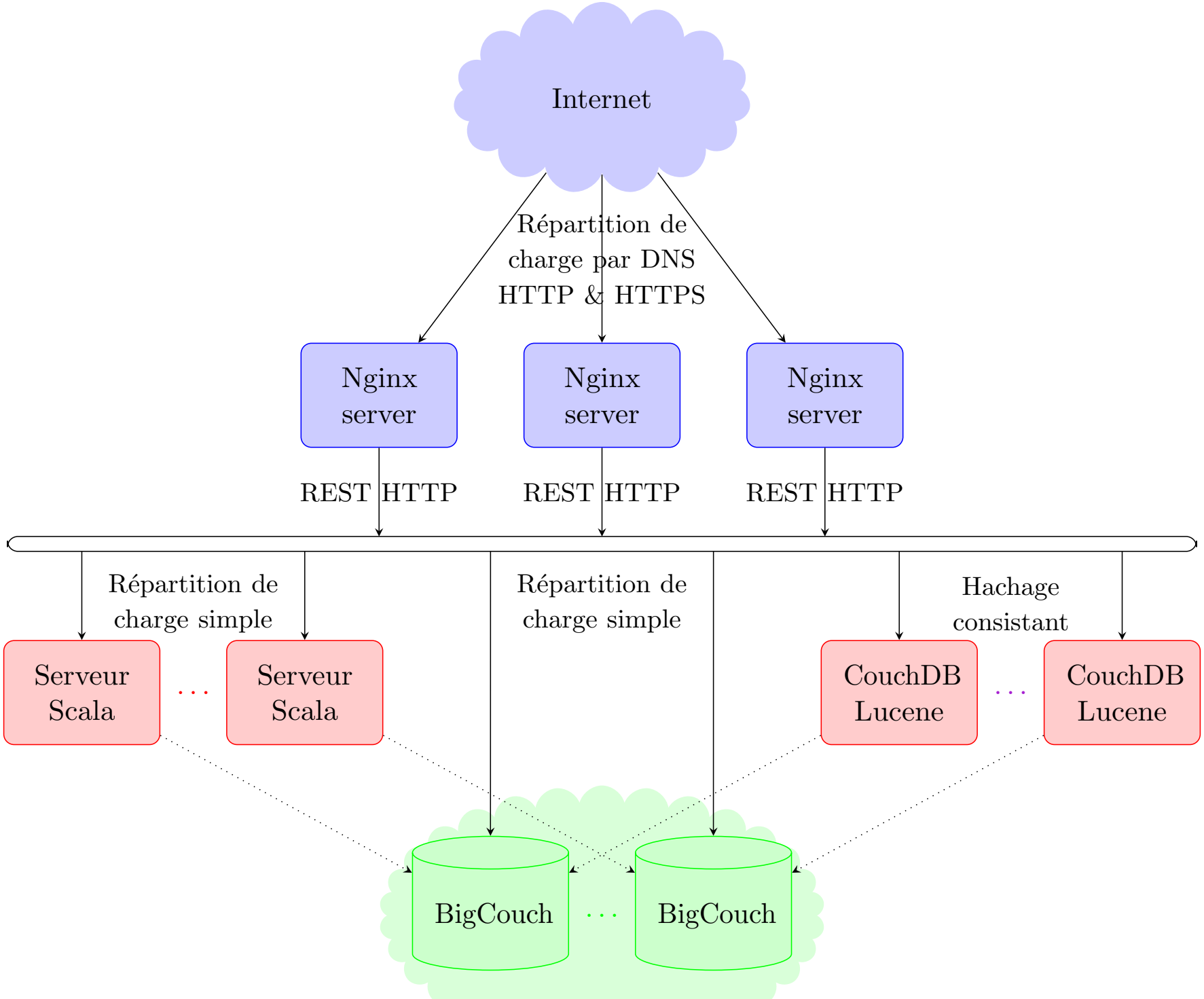

nooda_page_aaaa
nooda_page_temp
Nooda est une plateforme
⟶ réplication ⟶
Nooda est une plateforme
Nooda est une plateforme
mise à jour
Nooda est une plate-
forme de communication.
Le projet Nooda
est une plateforme
⟵ réplication ⟵
Conflit
Le projet Nooda
est une plateforme
résolution
Le projet Nooda est une
plateforme de communication.
Le projet Nooda
est une plateforme


nooda_page_aaaa
nooda_page_temp
Nooda est une plateforme
⟶ réplication ⟶
Nooda est une plateforme
Nooda est une plateforme
mise à jour
Nooda est une plate-
forme de communication.
Le projet Nooda
est une plateforme
⟵ réplication ⟵
Conflit
Le projet Nooda
est une plateforme
résolution
Le projet Nooda est une
plateforme de communication.
Le projet Nooda
est une plateforme

Plan

Architecture globale

- Client javascript :
  - site statique
  - communication avec les serveurs en AJAX (+COMET) via l'**API**
- Serveurs :
  - **Nginx** : porte d'entrée et serveur du site statique
  - **BigCouch** : base de données distribuée
  - **Scala** : serveur pour les requêtes complexes
  - **CouchDB-lucene** : serveur pour l'indexation


Internet
Répartition de
charge par DNS
HTTP & HTTPS
Nginx
server
Nginx
server
Nginx
server
REST HTTP
REST HTTP
REST HTTP
Répartition de
charge simple
Répartition de
charge simple
Hachage
consistant
Serveur
Scala
Serveur
Scala
CouchDB
Lucene
CouchDB
Lucene
BigCouch
BigCouch

Architecture détaillée

- BigCouch :
  - stockage des données
  - requêtes simples (via vues et *update handlers* en Javascript) avec versionning
- Scala :
  - requêtes *bulk*
  - vues complexes (descendants, ancêtres, ...)
  - requêtes d'aide à la gestion des conflits
  - création de pages
  - requêtes d'administration (*sanitize*)
- CouchDB-Lucene :
  - recherche

Plan

- **nooda_page_[id page]** :
  la page d'id [id page]
- **nooda_model** :
  la base de donnée à partir de laquelle, la page est crée
- **nooda_meta** :
  information sur les pages (date de création, ...)
  et site web d'administration

Organisation de CouchDB

Documents

- champs réservés à CouchDB (_*) :
  - **_id**, **_rev**
  - **_attachments** : contient les anciennes versions
  - **_conflicts**
  - ...
- champs réservés (*_) :
  - **parent_**, **type_** : obligatoires
  - **created_at_**, **updated_at_** : géré par serveur
  - **deleted_** : pour la gestion de version sur les éléments supprimés
  - ...
- autres champs (selon type) :
  - **name**
  - **content**
  - **previous**
  - ...

Plan

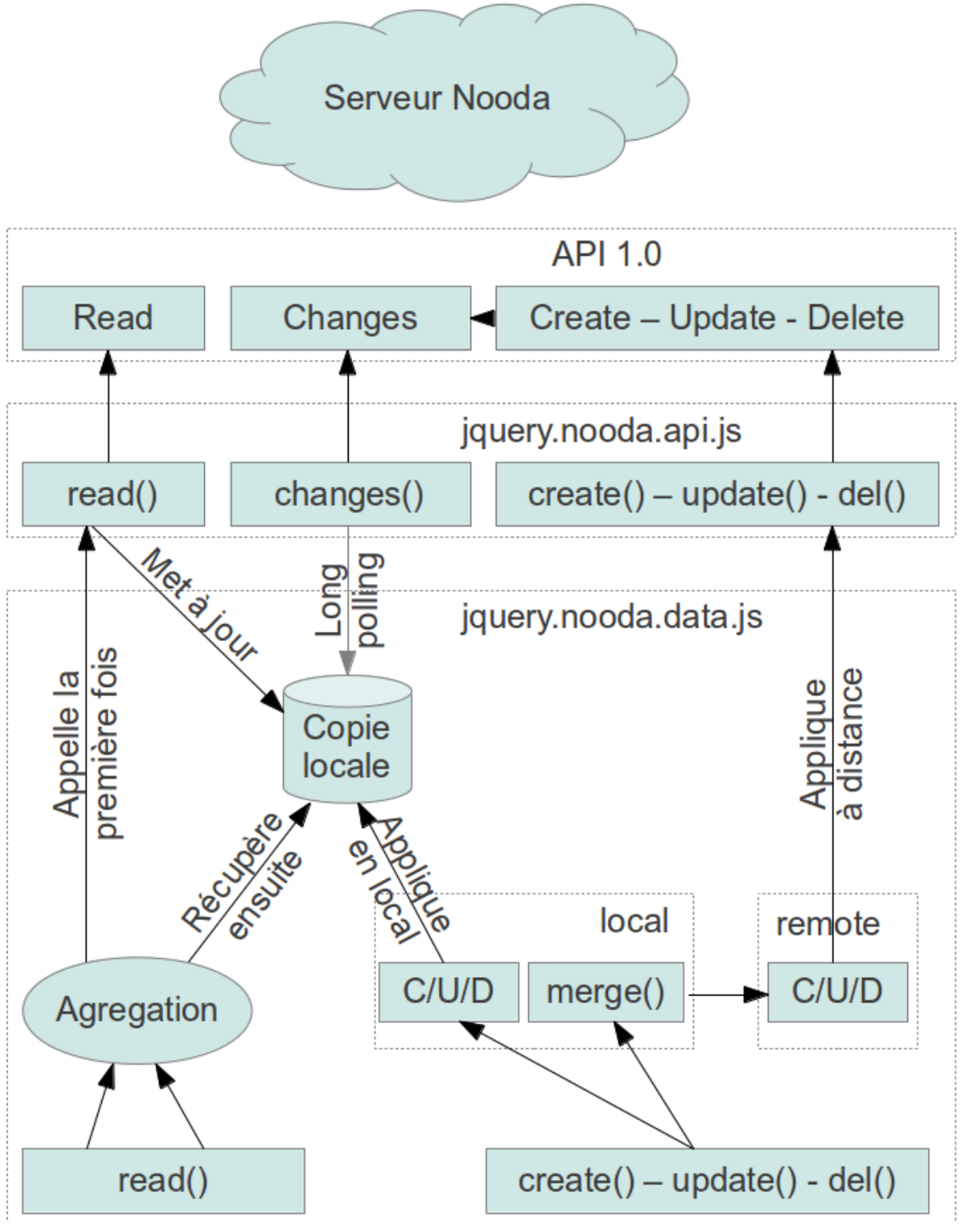
Gestion des versions et des conflits

La gestion de révisions de CouchDB ne doit être utilisé que pour la réplication.
Dans Nooda, chaque ancienne révision est stockée en fichier attaché aux documents.
Lors d'un conflit CouchDB, le serveur Scala est capable de retourner :

- les documents en conflits
- un ancêtre commun aux documents

Cela permet un *3-way merge*.

Plan


Serveur Nooda
API 1.0
Read
Changes
Create – Update - Delete
jquery.nooda.api.js
read()
changes()
create() – update() - del()
Met à jour
Long polling
jquery.nooda.data.js
Appelle la première fois
Copie locale
Applique à distance
Récupère ensuite
Applique en local
local
remote
Agregation
C/U/D
merge()
C/U/D
read()
create() – update() - del()

Deux librairies fournies par nooda !

- `https://nooda.net/res/api/jquery.nooda.api-0.4.js`
- `https://nooda.net/res/api/jquery.nooda.data-0.5.js`

Plan

Exemple d'utilisation : lecture et écriture

```
var page = $.nooda.api.page('5e87519bbe8b0225b4704d297e1a39f5');

// Get root items
page.children(null, function(data) {
  console.log(data.rows);
});

// Get all files
var data = page.descendants(null, null);
console.log(data.rows);

// Create a new folder
page.create({
  type_ : 'Collection',
  name : 'New folder',
  parent_ : null
}, {
  success : function(data) {
    console.log('Created ' + data.doc._id);
  },
  error : function() { console.log('Failed'); }
});
```

Plan

Exemple d'utilisation : lecture et écriture

```
var page = $.nooda.data.page('5e87519bbe8b0225b4704d297e1a39f5');

// Get root items
page.children(null, function(data) {
  console.log(data.rows);
});

// Get all files
var request = page.descendants(null, null);
console.log(request.data.rows);

// Create a new folder
page.create({
  type_ : 'Collection',
  name : 'New folder',
  parent_ : null
}, function(data) {
    console.log('Created ' + data.doc._id);
});
```

Exemple d'utilisation : événements et aggrégation

```
// Get root items
page.children(null, {

  aggregator : true,

  first : function(data) {
    console.log('Premiere fois');
  },

  each : function(data) {
    console.log('Chaque fois');
  },

  update : function(data) {
    console.log('A partir de la seconde fois');

    this.unbind(); // Arreter de recevoir
  }
});
```

Exemple d'utilisation : mode local et distant

```
var uuids = page.uuids(1, null);

// Create a new folder
page.create({
  _id : uuids[0],
  ...
});

// Create a new folder on server
page.remote.create({
  _id : uuids[0],
  ...
});

// Create a new folder on client
page.local.create({
  _id : uuids[0],
  ...
});

page.local.merge(uuids[0]);
```

Questions

Merci de votre attention

En attendant de vous voir nombreux sur

`https://nooda.net`