

# Ben, c'est l'histoire d'un MOOC...

## Retour d'expériences sur le SOOC « Programmation récursive »

C. Queinnec  
UPMC - LIP6



SIF - 23-24 juin 2014



# MOOC ? SOOC ?

- Massive (Small)
- Open
- Online
- Course

MOOC = Enseignement à distance + réseau social  
+ évaluation continue + élasticité

# Plan

- Le SOOC
- Quelques chiffres
- Déploiement
- Correction automatisée
- Perspectives et conclusions

<http://programmation-recursive.net/>

# Le SOOC

## Buts du SOOC

- Introduction à l'informatique (dans la lignée du SICP) pour débutants, lycéens, enseignants d'ISN
- Fondé sur un cours créé en 2000 et encore donné en 2013 à l'UPMC en L1 S1
  - Livre de cours, livre d'exercices, videos (2004 iTunes), bandes sons (mp3)
  - Nouveaux matériaux en CC-BY-NC-SA
- Doté d'exercices à correction automatisée
- Accent mis sur les tests (*Test Driven Development*)
- Avantages : récursion, Scheme, non-généré

## Calendrier

*Sem-38 : juin 2013* Encouragement UPMC, choix plate-forme et date de début : 18 février 2014

*Sem-6 : janvier 2014* Début filmage, début différé de 2 semaines

*Sem-2 : février 2014* Montage bande annonce, publicité

*Sem0 : fin février* Premières pages et vidéos prise en main

*Sem1 : 4 mars* Mise en ligne semaine 1 : bases

*Sem2 : 11 mars* Premier questionnaire

*Sem3 : 18 mars* Récursion linéaire

*Sem6 : 1er avril* Badges « maître en récursion linéaire »

*Sem7 : 8 avril* Récursion arborescente, second questionnaire

*Sem9 : 22 avril* Badges « maître récursion arborescente »

*Sem10 : 19 avril* Évaluation (texte → valeur)

*22 mai* Examen présentiel à Nice puis examen public en ligne et certificat

# Séquencement

- 10 semaines de cours
- parution tous les mardis matin (vidéos, transparents, exercices)
- et une brève dans le forum
- inscription non obligatoire (mais sans accès au forum ni aux exercices)
- ressources toujours en ligne (même si MOOC fini)

Contenu :

1. Récursion sur entiers naturels
2. Récursion sur listes
3. Récursion sur arbres
4. Processus d'évaluation (texte → valeur)

# Moyens

- 1 enseignant (0h)
- 4 moniteurs (environ 100h)
- 3 vidéastes (UPMC-CPM) pour environ 7 heures c'est-à-dire 60 vidéos
- de nombreux coups de main et discussions avec collègues, anciens collègues
- 2 machines chez OVH, 3 au LIP6, 3 à la PPTI et 1 chez moi
- x machines chez Google

**Merci à tous !**

# Caractéristiques

- exercices à correction automatisée appréciés au détriment des QCM
- vidéo-conférences en direct abandonnées au bout de 3 semaines par manque d'audience
- forum assez bien utilisé (et quelques courriels plus personnels)
- badges à portée limitée au forum
- certificat (attestation)



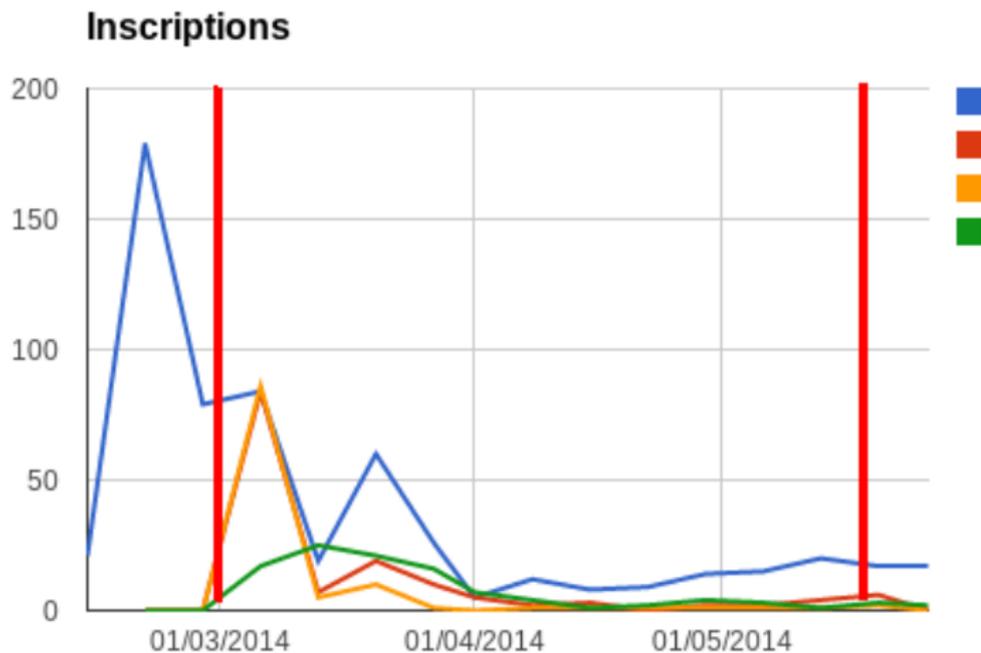


## *Leçons et remarques*

- cours de base déjà rôdé, matériel abondant
- chapitrage délicat
- nécessité d'une évaluation continue
- vidéos face à caméra
- monter une vidéo est un métier
- recruter une équipe pédagogique
- rétribution des auteurs de MOOC

# Quelques chiffres

# Nouvelles inscriptions/semaine

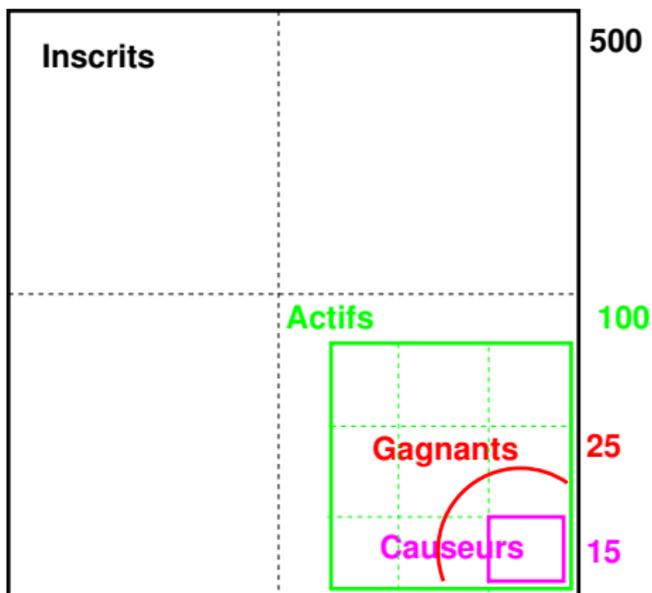


inscrits, avec nom-prénom, au forum, et au moins 1 exercice tenté.

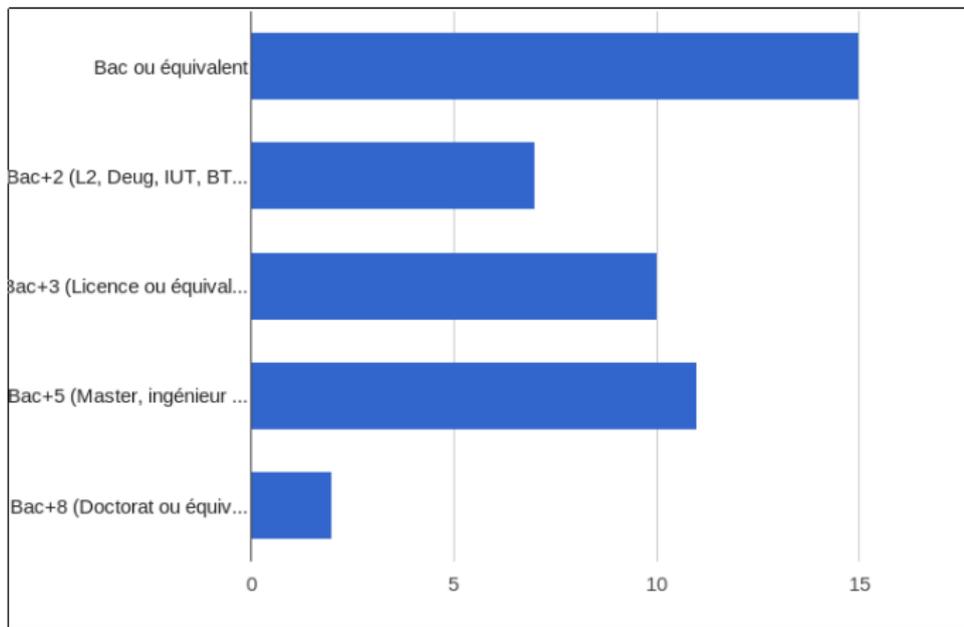
## Chiffres finaux (au 27 mai 2014)

inscrits	585
inscrits avec nom-prénom	144
inscrits au forum	110
intervenants au forum	13
sujets, messages sur forum	114,450
ayant tenté au moins 1 exercice	106
ayant tenté au moins 10 exercices	74
réponses au questionnaire 1	48
réponses au questionnaire 2	11
badgés récursion linéaire	14
badgés récursion arborescente	6
ayant réussi au moins la moitié des exercices à au moins à 80%	26
ayant tenté l'examen	25+10
certificats attribués	24

# Résumé des chiffres finaux (au 27 mai 2014)

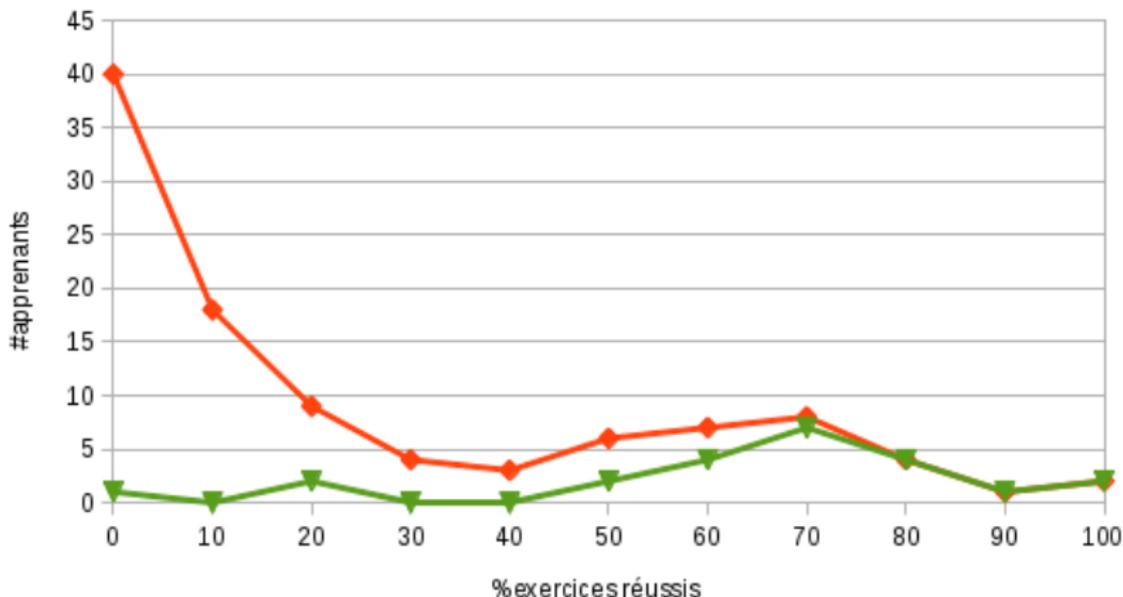


## Typologie questionnaire 1 (niveau d'étude)



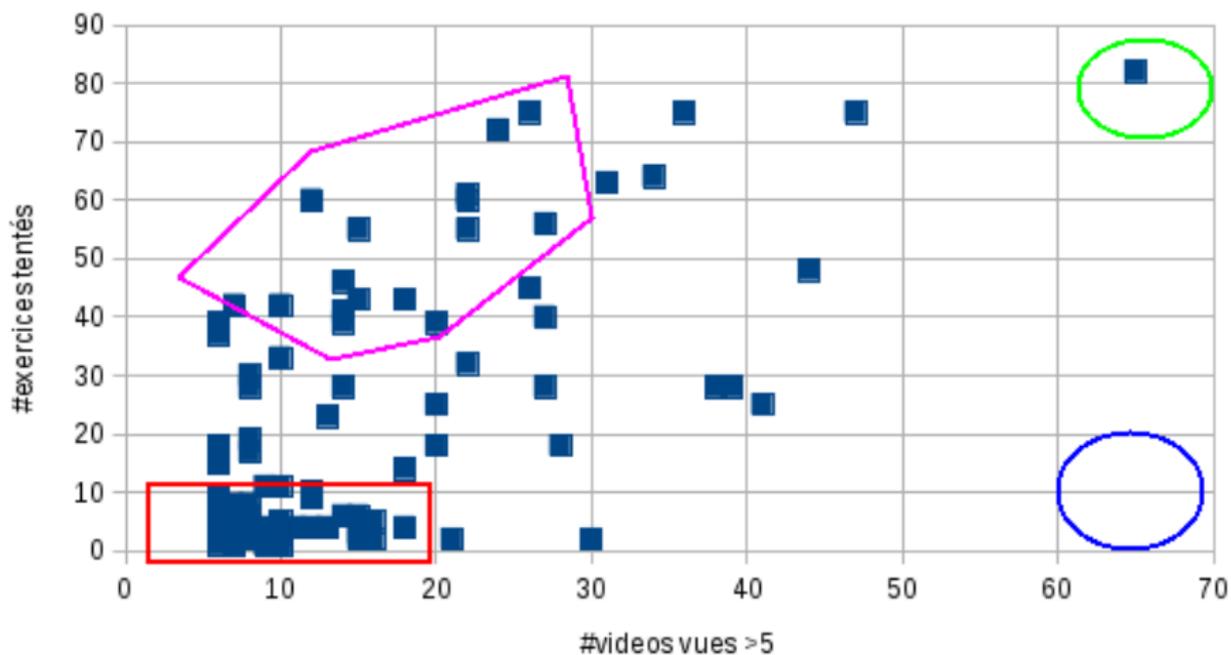
Oh combien de collègues...

## Réussite et % exercices tentés

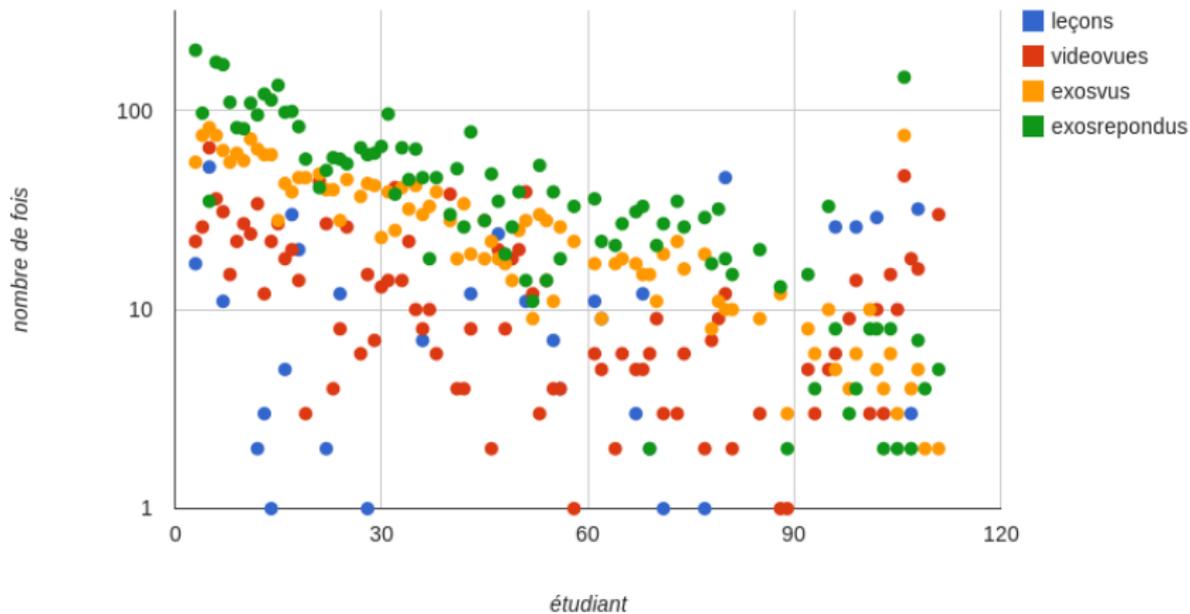


Histogramme sur les 102 apprenants ayant tenté au moins 1 fois un exercice. Seuls 35 apprenants ont tenté l'examen final, 24 l'ont réussi.

## Vidéos vues, exercices tentés



# Corrélations



Tri par nombre(log) d'interactions décroissant.



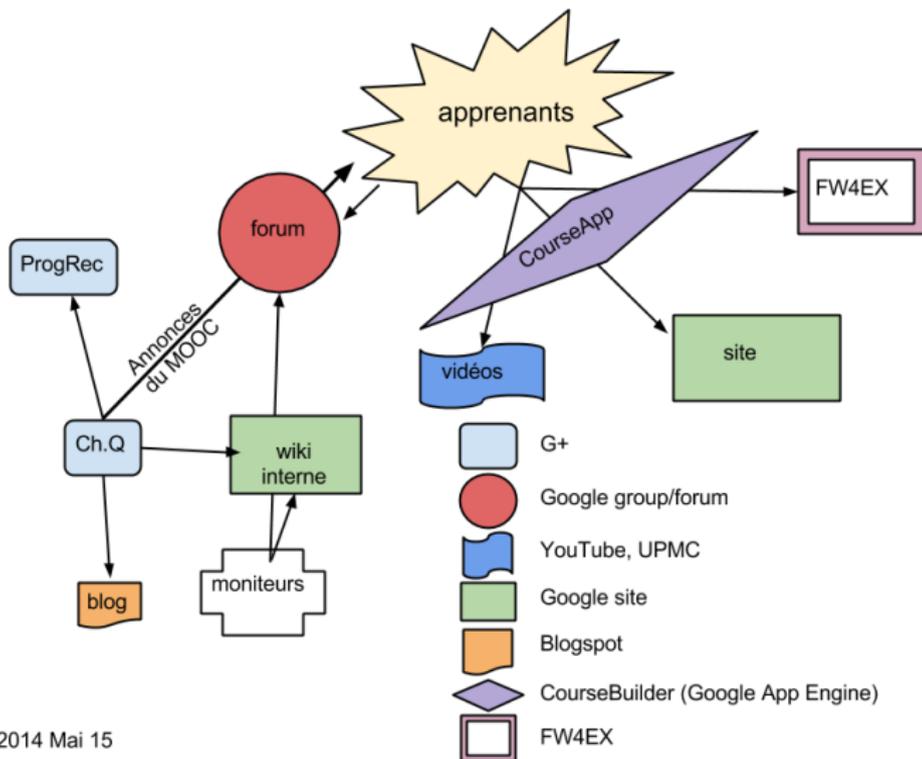
## *Leçons et remarques*

- Fusion de données
  - IP (Cache, Proxy(Squid), Tor)
  - TZ
  - téléchargement/*streaming*
- Jointures coûteuses
- Que chercher, que calculer ?
  - persévérance

# Déploiement

# De bric et de broc (merci Google)

La constellation du MOOC Programmation réursive



2014 Mai 15

# Déploiement

- Choix des canaux (G+, groupe=forum, courriel, etc.)
- Documents (CourseBuilder, site, site propre, etc.)

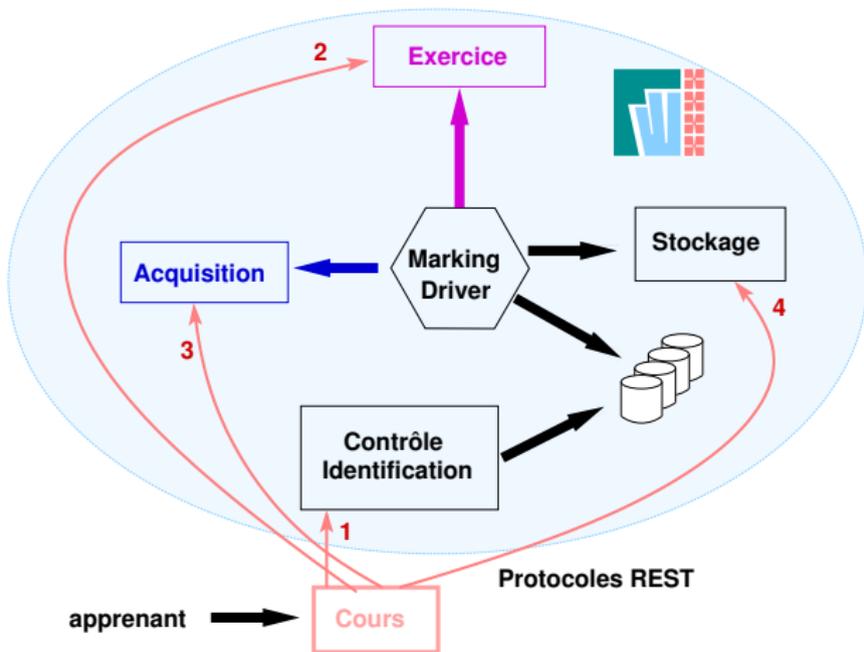
Les critères a priori

- Coût
- Facilité d'usage
- Aisance de déploiement
- 
- 

ou a posteriori :

Collecte de trace  
Archivage des contenus

# FW4EX – généralités



## *Du bonheur des machines virtuelles*

FW4EX se déploie maintenant au sein de machines virtuelles.

- déploiement facilité (libvirt, VMware)
- gel des versions des bibliothèques
- vitesse de rechargement (disques chaînés)

mais

- gros fichiers (4G compressés) :
  - temps de téléchargement,
  - mise à jour
- petits disques (12G) : gare aux scories !

## Exercice

Un exercice est juste un tar.gz, à déploiement aisé, contenant :

- un énoncé
- des scripts de correction
- des pseudo-copies

Les scripts sont confinés en fonctionnalités, en temps et en production d'octets.

Actuellement existent des exercices pour C, Java, bash, Octave, Scheme, OCaml.

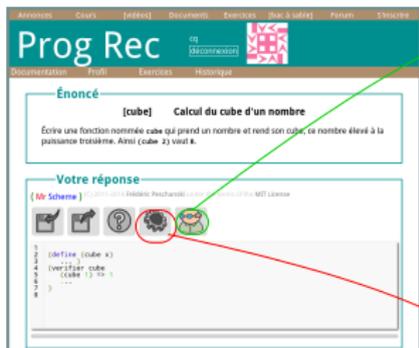
Plate-forme en fonctionnement depuis 2008 et 140 000 copies corrigées



Pour en savoir plus sur FW4EX « l'éternité pour vos exercices »

# Environnement de développement

## interprète Scheme distant (bigloo)



Prog Rec

Énoncé

[cube] Calcul du cube d'un nombre

Écrire une fonction nommée `cube` qui prend un nombre et rend son cube, ce nombre élevé à la puissance troisième. Ainsi (`cube 3`) vaut 8.

Votre réponse

(My Scheme) [?] [?] [?] [?] [?]

```
(define (cube x)
  (verifier-cube
   (cube 1) 0))
```



Rapport de correction automatique

Voici votre note maximale: 25 sur 100

Voici votre réponse:

```
(define (cube x)
  (let ((n 0))
    (cond [(= x 0) 0]
          [(= x 1) 1]
          [else 1])))
```

Voici ce que prévoit la correction automatique:

- Je vois lire votre fichier
- J'analyse le système de votre fichier
- J'exécute votre fichier
- Je passe avec succès la fonction `cube`
- Je teste la fonction `cube` contre tous vos 2 tests
- Je ne peux pas déterminer la fonction retournée par votre
- J'exécute mes solutions
- Je vois quand il me faut définir la fonction `cube`
- Je vois tester mes fonctions avec mes propres tests
- Ma fonction `cube` passe tous mes tests
- Je suis à présent que ma solution passe vos tests
- Ma fonction `cube` passe tous vos 2 tests
- Je suis maintenant que vos solutions passent mes propres tests

Je suis maintenant que vos solutions passent mes propres tests

Vous pouvez 25 points.



Votre réponse

(My Scheme) [?] [?] [?] [?] [?]

```
(define (cube x)
  (verifier-cube
   (cube 1) 0))
```

Évaluation du programme locale

2017-2018 Université de Montpellier - Université de Montpellier

Erreur de système: [Erreur de notation]



## Interprète Scheme local (en Javascript)

# Environnement de développement

- Langage restreint (en grammaire, mots-clés, bibliothèques)
- IDE local : MrScheme (interprète en Javascript) dû à F.Peschanski

The screenshot shows a web browser window with a navigation bar at the top containing links for 'Annonces', 'Cours', '[vidéos]', 'Documents', 'Exercices', '[bac à sable]', 'Forum', and 'S'inscrire'. The main header features the text 'Prog Rec' in large white font on a teal background, with a 'cq déconnexion' button and a red and white geometric pattern to the right. Below the header is a secondary navigation bar with 'Documentation', 'Profil', 'Exercices', and 'Historique'. The main content area is titled 'Énoncé' and contains the following text:

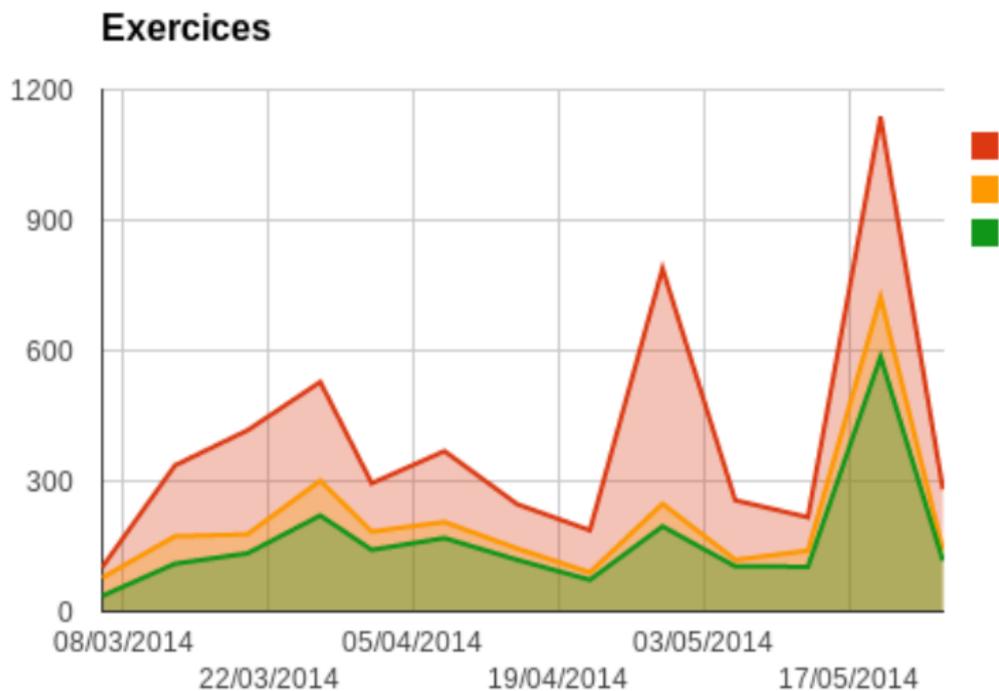
**[cube] Calcul du cube d'un nombre**

Écrire une fonction nommée `cube` qui prend un nombre et rend son cube, ce nombre élevé à la puissance troisième. Ainsi (`cube 2`) vaut `8`.

Below the problem statement is a section titled 'Votre réponse' which includes a copyright notice: '( Mr Scheme ) (C) 2011-2014 Frédéric Peschanski under the terms of the MIT License'. This section contains five icons: a folder, a document with an arrow, a question mark, a gear, and a person wearing glasses. At the bottom of this section is a text area with the following code:

```
1 (define (cube x)
2   ...)
3 (verifier cube
4   (cube 1) => 1
5   ...)
6 )
7
8
```

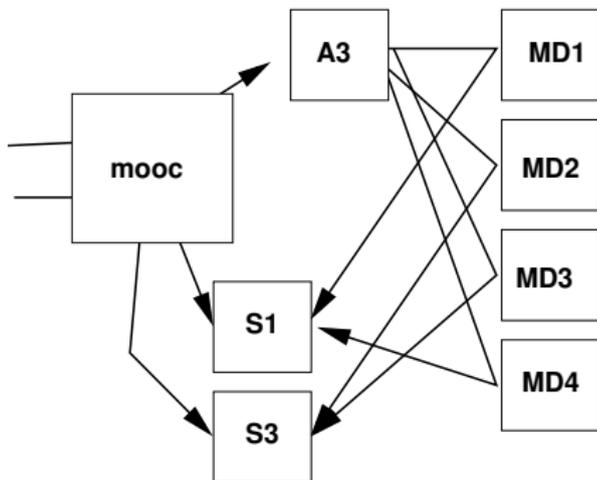
# Exercices / semaine



nombre d'essais, réussites à 80% ou plus, à 100%.

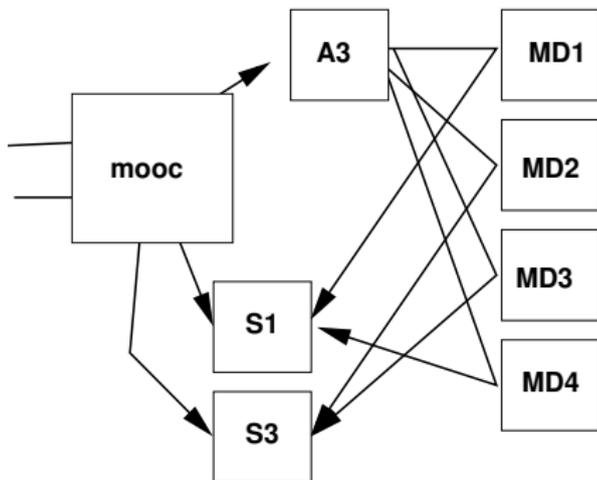
# Élasticité

- Pour vidéos, documents : plateformes spécialisées
- Pour FW4EX plus difficile à assurer
  - Charge proxys
  - Charge base de données
  - Charge correcteurs



# Élasticité

- Pour vidéos, documents : plateformes spécialisées
- Pour FW4EX plus difficile à assurer
  - Charge proxys
  - Charge base de données
  - Charge correcteurs



- et, finalement, n'a pas été nécessaire



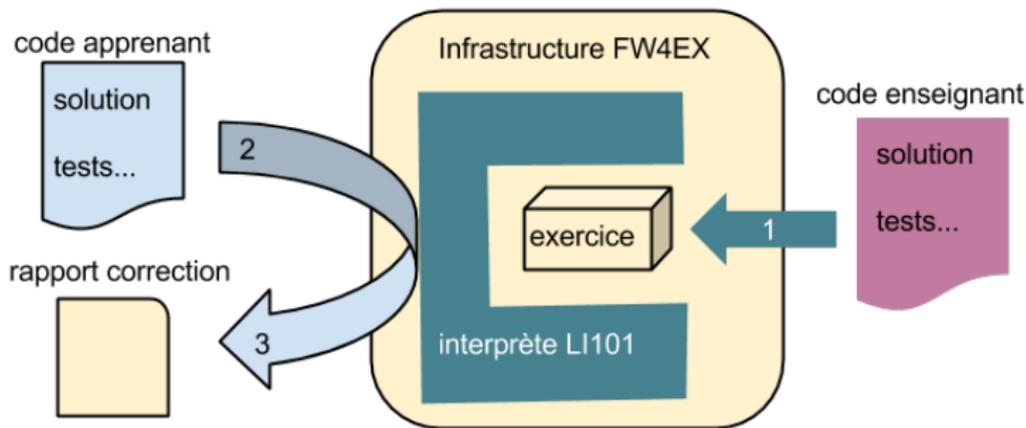
## *Leçons et remarques*

- Plateforme (Coursera, OpenEdX et autres LMS)
  - Déterminante pour la collecte des données
  - Déterminante pour les caractéristiques du forum
  - mais, qualités intrinsèques nécessaires :
    - déploiement automatisable (`makefile` par ex.)
    - greffe de code supplémentaire (hook, plugin, etc.)
    - extraction continue des données (csv, sql, etc.)
  - Niveau de collaboration (établissement, enseignant, etc.)
- Uniformité d'aspect

# Correction automatisée

## Les exercices du MOOC en FW4EX

- Tests unitaires par comparaison (environ 15 secondes)
- Tests progressifs
- Verbalisation des tests



## Notation

Le langage étant fonctionnel, on demande des fonctions.  
L'apprenant écrit (et l'auteur de l'exercice itou) :

```
(define (foo ...)           $f_s$ 
  ... )
(verifier foo               $v_s$ 
  (foo ...) => ...        ; au moins 2 tests
  (foo ...) => ... )
```

1. cohérence(0% éliminatoire) : on vérifie  $v_s(f_s)$
2. correction1(0% (Ouf!)) : on vérifie  $v_s(f_t)$
3. correction2(50%) : on vérifie  $v_t(f_s)$
4. couverture(50%) : on compare  $v_s(f_s)$  et  $v_t(f_s)$

## *Notation par comparaison*

- Suite de tests unitaires
- Les fonctions de l'enseignant doivent être précautionneuses c'est-à-dire vérifier que les hypothèses sont bien respectées
- Tests conjoints sur barrière d'abstraction ou comment tester plusieurs fonctions devant être, ensemble, cohérentes
- Délicate séparation entre l'espace des fonctions étudiant et enseignant : une fonction peut en cacher une autre !

# De l'importance de la verbalisation

## ☑ Rapport de correction automatique ☹

Voici votre note normalisée: **25** sur 100

Voici votre réponse:

```
1
2 (define (cube x)
3   (* x x) )
4 (verifier cube
5   (cube 1) => 1
6   (cube 1) => 1
7
8 )
9
```

Voici ce que produit la correction automatisée:

- ✓ Je vais lire votre fichier
- ✓ J'analyse la syntaxe de votre fichier
- ✓ J'évalue votre fichier
- ✓ Vous avez défini la fonction cube
- ✓ Votre fonction cube passe tous vos 2 tests
- ✓ Je lis maintenant le fichier contenant ma solution
- ✓ J'évalue ma solution
- ✓ J'ai, quant à moi, défini la fonction cube
- ✓ Je vais tester mes fonctions avec mes propres tests.
- ✓ Ma fonction cube passe tous mes 4 tests
- ✓ Je teste à présent que ma solution passe vos tests
- ✓ Ma fonction cube passe tous vos 2 tests
- ✓ Je teste maintenant que vos solutions passent mes propres tests
- ✗ **Votre fonction cube échoue sur mon test numéro 2 et voici le test en question: (cube 2) Je m'arrête là**

Vous gagnez **25** points.



## *Leçons et remarques*

- Interprète instrumenté pour le langage enseigné
  - identité entre IDE de développement et de notation
- Susciter, maintenir l'intérêt
  - tableau de chasse
  - exercices graphiques et galerie associée
  - badges sur forum
  - certification finale
- Faciliter la progression (montrer test défaillant)
- Côté enseignant :
  - suivre la progression des apprenants
  - faciliter la vision des copies
  - bon pour l'ego !

# Perspectives et conclusions



## *Améliorations à apporter*

- promouvoir, soutenir, inciter continûment
  - courriel hebdomadaire, bilan de la semaine
  - organiser des débats, des partages de code
- retour sur exercice réussi (et problème de divulgation des solutions)
- regrets (entre autres) :
  - binômage (duo-duel)
  - *epsilon-better peeping*
- analyse des principaux types d'erreur
  - amélioration des correcteurs
  - prise en compte de la complexité
  - vérification du style



## *Nouvelle édition ?*

Au cas où il y aurait une seconde édition, de nombreuses choses à modifier :

- certaines vidéos à scinder, refaire, enrichir
- ajouter index, glossaire, références
- plus d'exercices, plus progressifs, mieux indexés
- des rapports de correction d'exercice plus étoffés
- utiliser le facteur multiplicatif (binôme, classification des réponses, etc.)

MOOC cherche parrainage !



## *Perspectives*

- Enfin la possibilité d'expériences pédagogiques significatives !
- Expérience enrichissante mais obsédante
- Préparation d'un MOOC « Correction Automatisée de Programmes Informatiques »

## Questions ?

- Après-midi « MOOCs à l'UFR d'ingénierie » avec Fabrice Kordon et Christian Queinnec : 7 juillet, amphitheâtre 25, UPMC Campus Jussieu.
- « Monter un MOOC de A à Z » de Matthieu Cisel *et alii* sur FUN
- *Les MOOC, Conception, usages et modèles économiques*, J.C. Pomerol, Y. Epelboin, C. Thoury, Dunod 2014

