

Vers un **exerciseur adaptatif** pour **apprendre la programmation en autonomie**

Programmer's Learning Machine

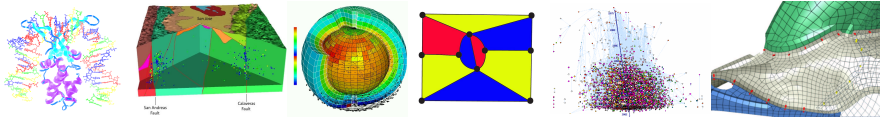
Martin.Quinson@loria.fr
Équipe-projet Algorille, Nancy

Journée MOOC Lab
13 janvier 2014

Computational Science *of* Computer Systems

Computational Science

Complex models to understand and predict the studied phenomena



Large-Scale Distributed Systems

- ▶ Complex hierarchies : grid, cluster, multicore
- ▶ Massive parallelism : 10^6 cores (10^9 in 2020)
- ▶ Heterogeneity (GPU, SOC) ; Dynamicity



Scientific Challenge: Study system **correctness** and **performance**

- ▶ Need for adapted **scientific instruments**, combining approaches
- ▶ **Modeling** to understand system dynamics ; **Simulation** to predict experiments

SimGrid scientific instrument

- ▶ Versatile, validated, usable (dozens papers per year by hundreds)
- ▶ Full-featured model-checker toward formal verification of MPI applications

Mon expérience d'enseignant

L'autre moitié du temps

- ▶ Enseignant depuis ma maîtrise, depuis 2005 à Telecom Nancy
- ▶ D'abord sur mes recherches, mais niveau des élèves faible en programmation
- ▶ Depuis 2006 : bases de l'aglo-prog (algo, Java, scala, C, shell, systèmes)

Constat : seuls ceux qui savent programmer avant savent après

- ▶ Les autres ont des problèmes avec *if* au bout de 6 mois !

Comment aider ceux qui en ont besoin ?

- ▶ Énormes différences de niveaux en entrée : ceux qui savent trustent la parole
- ▶ Expliquer les bases de base : aussi difficile qu'enseigner à faire du vélo
- ▶ *Si seulement ils pouvaient bosser par eux-mêmes. . .*
- ▶ *. . . si seulement nos TP pouvaient être moins arides et plus attirants*

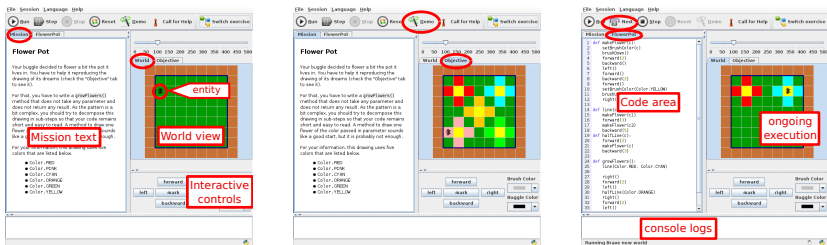
PLM : Programmer's Learning Machine (2008)

- ▶ Développé (avec G. Oster) pour une mise à niveau pseudo-autonome tutorée

PLM aujourd'hui

Exerciseur interactif dédié à la programmation

- ▶ Outil interactif et graphique pour apprendre à coder
- ▶ C'est en forgeant qu'on devient forgeron (et qu'on apprend à aimer ça)



Usage classique

- ▶ On lit la mission à gauche, on compare à droite l'état initial et l'état désiré
- ▶ On tape le code, on clique sur un bouton, et ça s'anime à droite
- ▶ Boucle de feedback très courte (et motivante pour les élèves)

Bonnes propriétés de PLM

- ▶ L'interface est bilingue anglais/français (soon italien ?)
- ▶ Les exercices sont trilingues Java/Scala/Python (+javascript/ruby ?)
- ▶ Mode démo, exécution pas-à-pas, vitesse d'animation pas à pas, sessions
- ▶ Plusieurs mondes parallèles pour mieux tester le code élève
- ▶ Documentation embarquée dans l'outil

L'univers des buggles

- ▶ **Bouger** : avancer, tourner, reculer. Getter/setter de position et de direction
- ▶ **Buggle** : getter/setter de couleur, de couleur de brosse, lever/baisser
- ▶ **Interactions** : Lire la couleur de la case ; Lire / écrire du texte par terre
- ▶ **Baggles** : Des biscuits qu'on trouve par terre, et qu'on peut ramasser
- ▶ **Murs** : tester leur présence ; ils sont infranchissables

D'autres univers

Différents types d'univers pour différentes situations-problèmes

L'univers des tortues

- ▶ Moins d'interaction avec le monde, mais virages au degré près
- ▶ Avance, recule, tourne, lève/baisse crayon, getter/setter d'angle et position
- ▶ C'est le logo classique, toujours efficace pour la récursivité

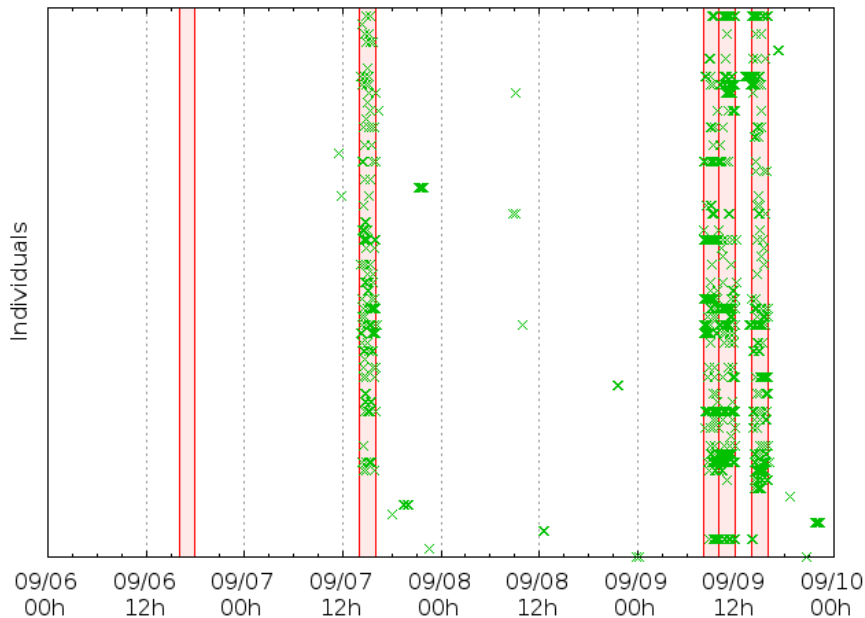
L'univers des tris

- ▶ Un tableau, auquel on accède par les builtins uniquement
- ▶ Décompte exact des accès et comparaisons \leadsto identification de l'algo de tri
- ▶ Une vue temporelle pour voir (et comparer) la chronologie de l'algorithme

Encore plus d'univers

- ▶ Drapeau hollandais, Crépier, Baseball multicolor : variation sur les tris
- ▶ Bat : tests simples de fonctions, à la jUnit
- ▶ Hanoï : les tours ; Lightbot : jeu / casse tête
- ▶ Ajouter un univers : 100 à 300 lignes de Java

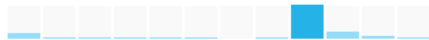
Nos élèves travaillent par eux-mêmes



Des gens utilisent l'environnement

Dans twitter

2013



2012



2011



2010

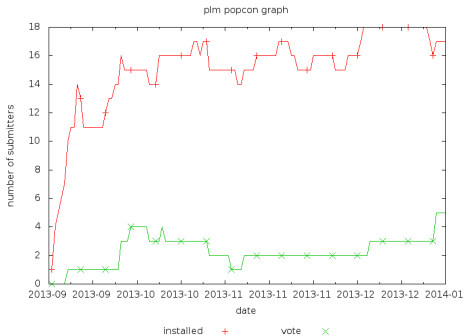


2009



Sep 2013: 3,958 Tweets

Dans Debian



Mais je ne sais pas !

Notre progression pédagogique en CS0

Objectif : tactical programming

- ▶ Le B-A-BA de la programmation : écrire des trucs de base sans difficulté
- ▶ Que ce ne soit plus un problème quand les algos seront non-triviaux

Implémentation language-agnostic (Java, Python, Scala)

- ▶ Instructions (et commentaires)
- ▶ Conditionnelles (+40 exercices Bat pour savoir écrire des tests logiques)
- ▶ Boucles while ; switch cases (+ 3 exos d'applications)
- ▶ Les variables
- ▶ Boucles for et do/while
- ▶ Méthodes (+5 exos qui forcent à factoriser le code)
- ▶ Méthodes avec résultat, avec paramètres
- ▶ Tableaux (+12 exos d'applications, certains en Bat)
- ▶ Application : parcours de tableau 2D (5 exos) les turmites (turing 2D – 4 exos)

160 exos scénarisés, pour plus de 40h de bonheur

C'est même documenté dans PLM directement

	Instructions	Comments	Conditionals	While loops	Variables	For loops	Do/While loops	Methods	Switch	Arrays
Welcome in the Baggles' World		✗	✗	✗	✗	✗	✗	✗	✗	✗
Java Instructions		✗	✗	✗	✗	✗	✗	✗	✗	✗
Writing more complex programs			✗	✗	✗	✗	✗	✗	✗	✗
Conditional instructions	✓	✗		✗	✗	✗	✗	✗	✗	✗
While loops	✓	✗	✓		✗	✗	✗	✗	✗	✗
Baggle Seeking	✓	✗	✓		✗	✗	✗	✗	✗	✗
Storing and manipulating data	✓	✗	✓	✓		✗	✗	✗	✗	✗
For loops	✓	✗	✓	✓	✓		✗	✗	✗	✗
Do .. while loops	✓	✗	✓	✗	✗	✗		✗	✗	✗
Methods	✓	✗	✓	✓	✗	✗	✗		✗	✗
	Instructions	Comments	Conditionals	While loops	Variables	For loops	Do/While loops	Methods	Switch	Arrays
Building methodically	✓	✗	✗	✗	✗	✗	✗		✗	✗
Methods returning a result	✓	✗	✓	✗	✓	✓	✗		✗	✗
Methods with parameters	✓	✗	✓	✗	✗	✓	✗		✗	✗
Methodically drawing	✓	✗	✗	✗	✗	✗	✗		✗	✗
Methodically drawing (only bigger)	✓	✗	✗	✗	✗	✓	✗		✗	✗
Drawing bigger and bigger	✓	✗	✗	✗	✗	✓	✗		✗	✗
Even more pattern to draw	✓	✗	✗	✗	✗	✓	✗		✗	✗
Baggle Dance Revolution	✓	✗		✓	✗	✗	✗	✗	✗	✗
Baggle Dance Revolution 2	✓	✗		✓	✗	✗	✗	✗		✗
Slug Hunting	✓	✗	✓	✓	✗	✗	✗	✓	✗	✗
Slug Tracking	✓	✗	✓	✓	✗	✗	✗	✓	✗	✗
	Instructions	Comments	Conditionals	While loops	Variables	For loops	Do/While loops	Methods	Switch	Arrays
Snake World	✓	✗	✓	✓	✓	✗	✗	✓	✗	✗
Knitting and Arrays	✓	✗	✓	✗	✗	✓	✗	✓	✗	
Knitting, Arrays and modulus	✓	✗	✓	✗	✓	✓	✗	✓	✗	
Traversal by column	✓	✗	✓	✓	✓	✗	✗	✓	✗	
Traversal by line	✓	✗	✓	✓	✓	✗	✗	✓	✗	
Zig-zag traversal	✓	✗	✓	✓	✓	✗	✗	✓	✗	
Diagonal Traversal	✓	✗	✓	✓	✓	✗	✗	✓	✗	

Malheureusement, ceci n'est pas généré automatiquement (pas toujours à jour)

Point de vue d'enseignants

Usages en production

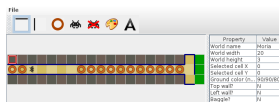
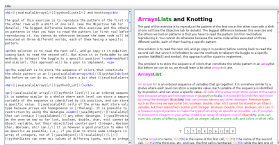
- ▶ Télécom Nancy : utilisé dans plusieurs modules (même pendant ma délég)
- ▶ Beaucoup d'intérêt, quelques usages disparates chez les collègues

Avant de l'utiliser chez vous

- ▶ Utiliser un tel outil demande un **gros travail de préparation**
Tester en salle de TP, faire les exos, anticiper les questions
- ▶ Un plan B en cellulose pour la première fois, mais c'est plaisant après

Créer des ressources (exos, leçons et univers)

- ▶ Ajouter des ressources très simple, mais nous sommes les seuls à l'avoir fait :-/
- ▶ Un mode "édition" est en travaux (papier à musique des exo de prog)



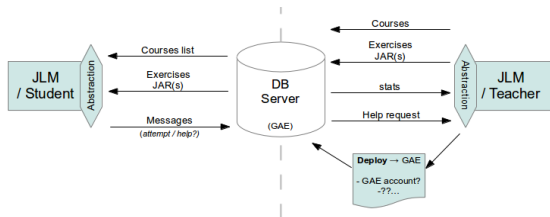
PLM et le métier d'enseignant

«Qui a besoin de moi ??»

- ▶ PLM libère le temps d'aider ceux en difficulté, mais on ne sait plus qui c'est

PLM espionne vos élèves

- ▶ Poste les bonnes nouvelles dans twitter (désactivable)
- ▶ Mais cela n'indique que ceux qui **n'ont pas** besoin d'aide
- ▶ Il faudrait un serveur capturant chaque tentative...



Projet de recherche autour de PLM

Participants potentiels

- ▶ M. Granbastien : Pédagogie et numérique (professeur émérite)
- ▶ A. Brun et A. Boyer : User modeling and personalization (Kiwi)
- ▶ Y. Toussaint : Knowledge Extraction from Texts (EPI Orpailleur)
- ▶ G. Oster : Computer-Supported Cooperative Work (EPI Score)
- ▶ M. Quinson : modélisation sémantique de systèmes distribués (EPI Algorille)

Objectif : PLM comme terrain d'expérimentations

- ▶ **Pédagogie** : quelle est la bonne façon d'enseigner la programmation ?
- ▶ **Modélisation** : gamification, tuteur intelligent
- ▶ **Data-mining** : erreurs corrélées, corrélation au background des élèves, erreurs typiques (feedback spécifique aux obstacles didactiques)
- ▶ **Collaboratif** : susciter la collaboration sans triche, proposer des exos
- ▶ **Systèmes distribués** : former suffisamment d'ingénieurs MPI pour l'avenir, diffuser SimGrid

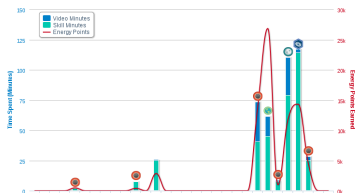
Monitorer les apprenants pour les aider

- ▶ Les médecins consultent le monitoring des patients pour leur sauver la vie
- ▶ Les enseignants **travaillent en aveugle** et demandent aux apprenants

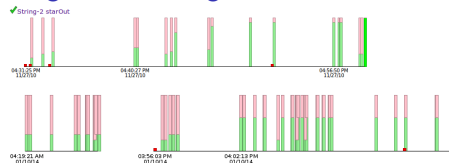
Activité sur Khan Academy

TristanQuinson Make a recommendation

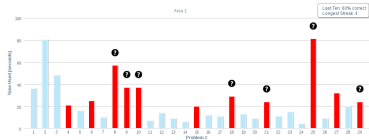
Skills Recommendations Videos Badges Activity



Progrès sur CodingBat



Problèmes sur un exo (Khan)



Détail du problème (Khan)

The screenshot shows the Khan Academy interface for the 'Area 1' problem. The problem text is: "Rectangle area as product of dimensions (same as counting unit squares)". The question is: "What is the area of the rectangle?". The user has input '7' for the width and '3' for the height. The answer field shows '21 square units'. The interface includes navigation buttons like 'Previous Problem', 'Next Problem', and 'Check Answer'.

Conclusions

PLM veut aider trois publics

- ▶ **Élèves** : apprennent à leur rythme dans un “jeu” sérieux
- ▶ **Enseignants** : simplifie la menée d'exercices, monitoring d'élèves à aider
- ▶ **Auteurs** : réutilisation du code non-fonctionnel, retour par instrumentation
- ▶ (plus les chercheurs sur ces thèmes)

Pleins de petites idées pour la suite

- ▶ **De nouveaux univers** : backtracking, POO, système, PatchWorld, Robbozzle
- ▶ **De nouveaux langages** : Tous les JSR231 et même le langage C ou scratch
- ▶ **Polishing** visuel, fonctionnel et documentation

Mais aller vraiment plus loin est un **projet de recherche**

- ▶ **Multidisciplinaire** (tradition de didactique de l'info à Nancy)
- ▶ **Balbutiant** voire tatonnant un peu
- ▶ **Mais passionnant!** yapuka