

Elèves-ingénieurs

Julian Bilcke

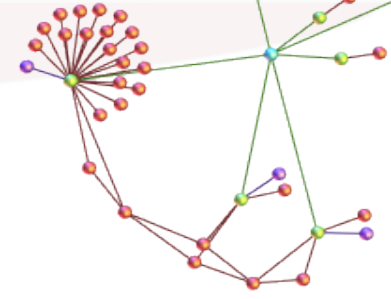
Sébastien Heymann

Microscope numérique de logiciels

"Network Dynamics"

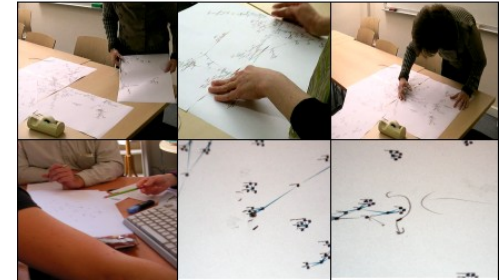
Suiveur UTC

Mr Stéphane Crozat



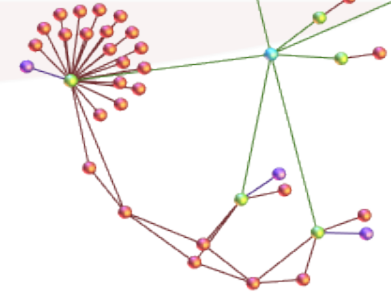
Présentation

- Approche issue des sciences humaines : infoVis, cartographie de l'information
- Rétro-ingénierie pour la représentation
- Limites de l'UML
- Public visé : architectes et développeurs de logiciels



Comment rendre compte de la nature des interactions entre méthodes d'un programme ?

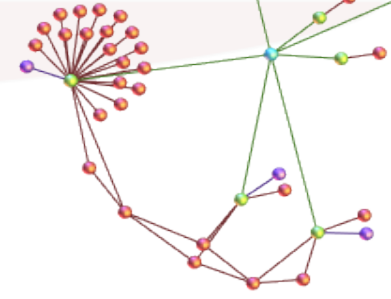
Notre proposition : les représenter sous forme de graphes de dépendances



Plan

1. Historique
2. Objectifs
3. Technologies
4. Réalisation
5. Bilan et perspective





Historique

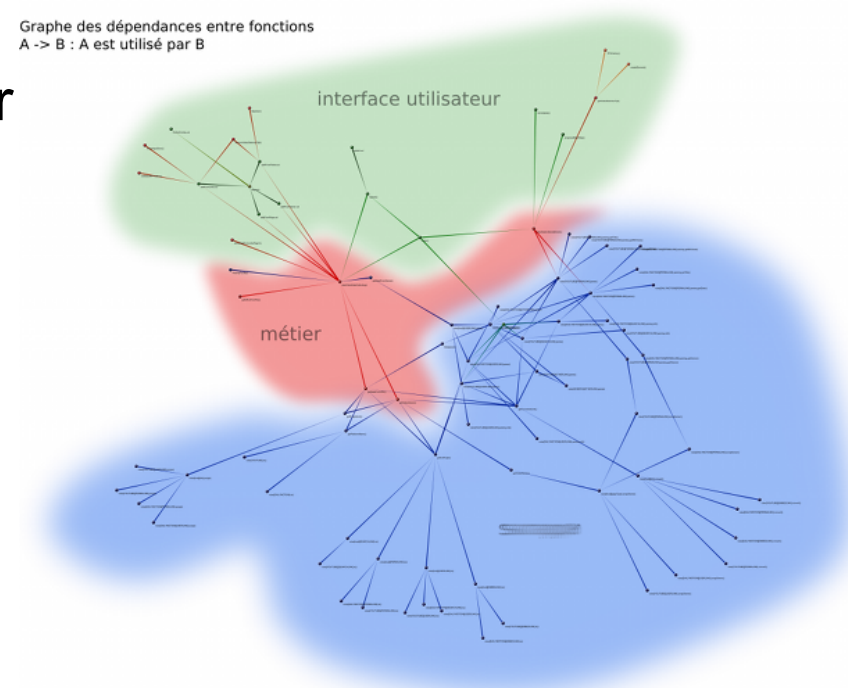
- Printemps 08 : UV TX Heymann-Palmier
- Analyse statique de logiciels Java

Actuellement

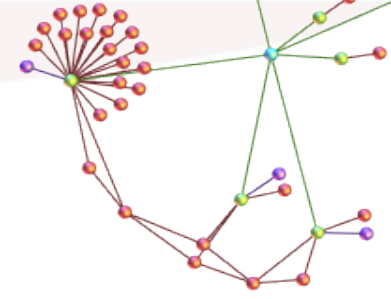
- Une PR : J.Palmier (analyse statique)
- Une TX : analyse dynamique

Structure de ~~XXXXXXXXXX~~

Graphe des dépendances entre fonctions
A -> B : A est utilisé par B



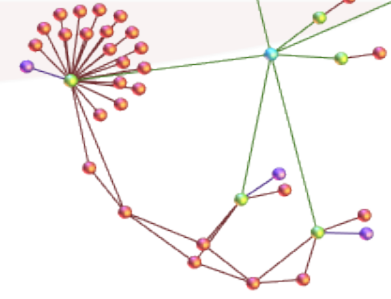
Spatialisé avec l'algorithme Fruchterman-Reingold depuis une analyse manuelle du programme



Objectifs opérationnels à terme

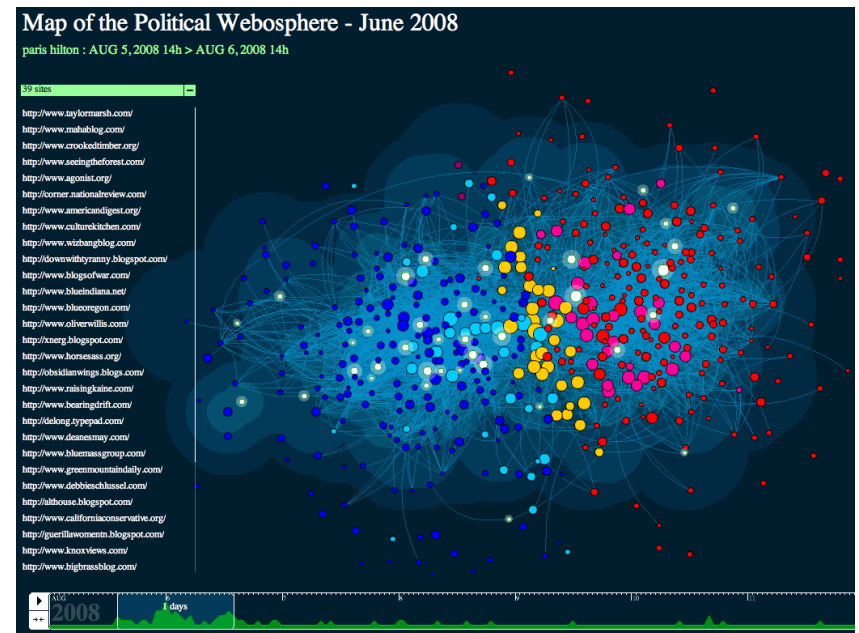
- **Formation** : intégration accélérée d'un nouveau développeur grâce à une carte du code source (plan de développement de sa « région », frontières, voisins)
- **Débogage** : recherche des modifications dans la structure du code facilitant la mise en place d'une stratégie de correction limitant les régressions
- **Veille** : vue sur un « panier de classes »

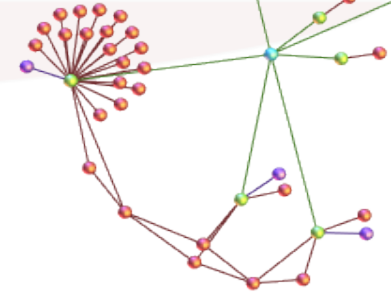




Objectifs TX

- Relever le challenge de l'analyse dynamique
- Source de données : dépôt SVN
- Outil : interface d'exploration temporelle (timeline et graphe différentiel)





Choix de représentation

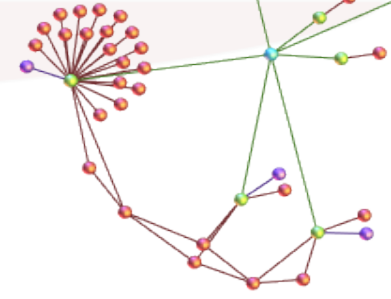
Que représenter dans la structure d'un programme Java (paradigme objet) ?

Éléments

- packages
- classes
- méthodes
- attributs de classe

Relations

- dépendances d'utilisation (appels) entre méthodes
- parenté (arbre AST)
- héritages
- implémentations d'interface



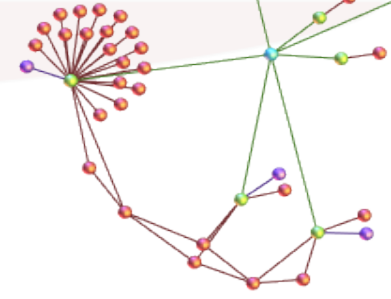
Technologies

- Plateforme Eclipse PDE
- Parseur JDT (Eclipse) étendu à nos besoins
- Bibliothèque JavaHL (plugin Subclipse) pour la manipulation des SVN

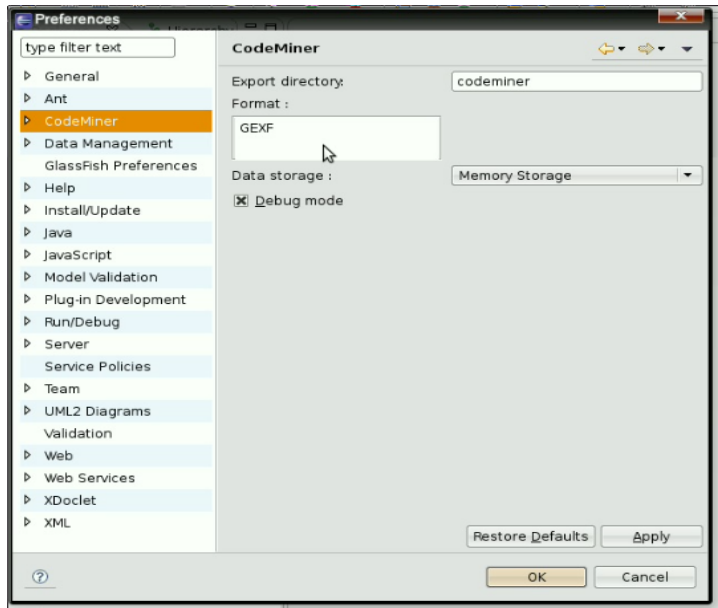
- Logiciel d'exploration et de manipulation de graphes Gephi (association Gephi)



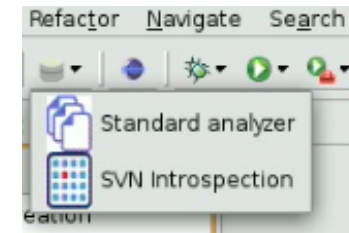
=> Code léger, robuste et évolutif



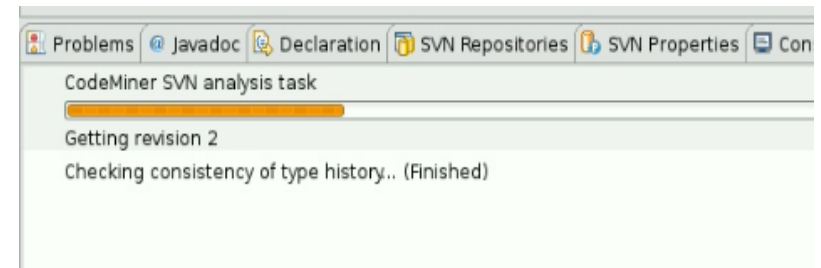
Réalisation : plugin Eclipse



- Configuration du format

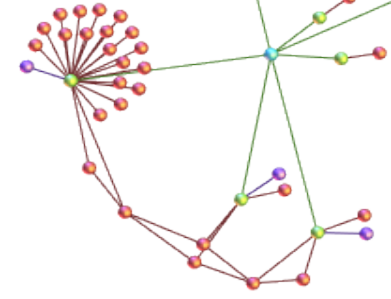


- Choix du mode d'analyse



```
<?xml version="1.0" encoding="UTF-8"?>
<gexf version="1.0" xmlns="http://www.gephi.org/gexf">
  <meta>
    <creator>codeminer</creator>
  </meta>
  <graph type="dynamic">
    <attributes class="node" type="dynamic">
      <attribute id="0" title="type" type="string"/>
      <attribute default="0" id="1" title="codelines" type="integer"/>
      <attribute id="2" title="modifier" type="string"/>
      <attribute id="3" title="comment" type="string"/>
    </attributes>
    <attributes class="edge" type="dynamic">
      <attribute default="dependancy" id="4" title="relationtype" type="string"/>
    </attributes>
    <nodes>
      <node id="0" creator="codeminer" label="hin" type="type" weight="1" x="0" y="0" z="0" />
    </nodes>
  </graph>
</gexf>
```

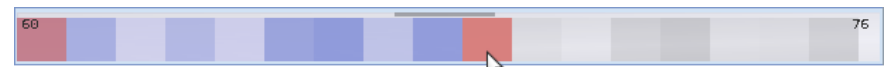
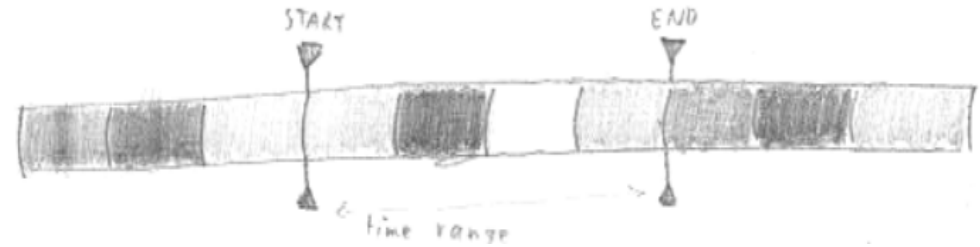
- Génération du GEXF



Réalisation : plugin Gephi

- **Etapes de la conception :**

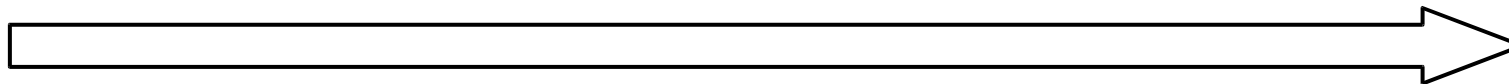
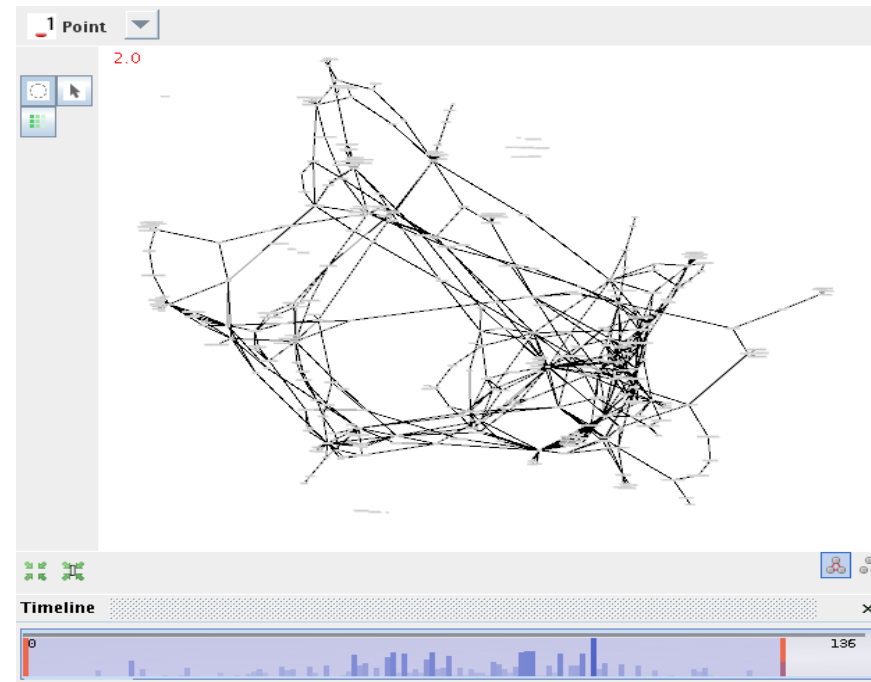
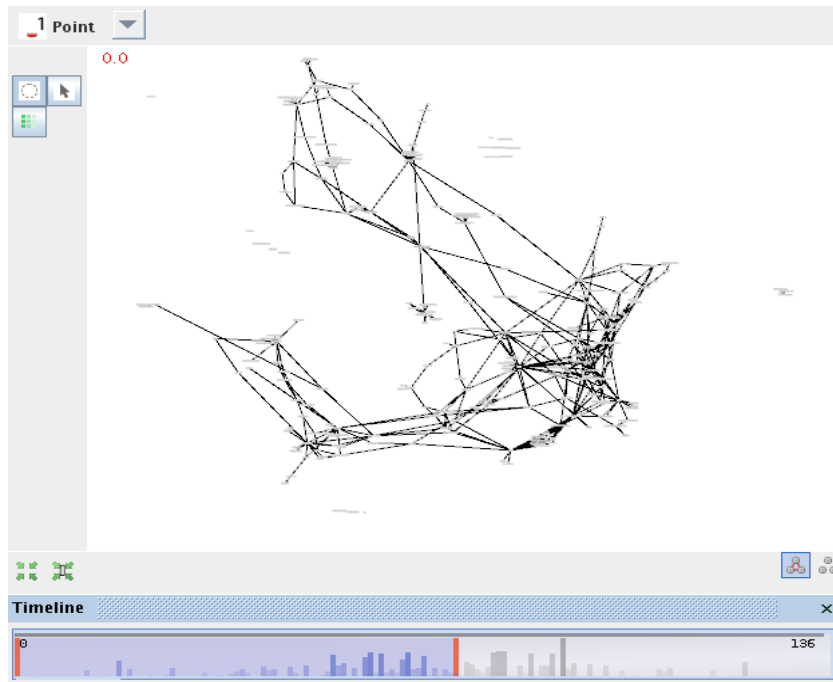
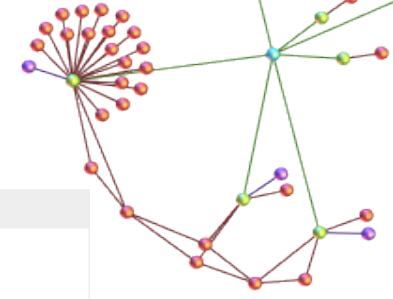
- Premières idées et croquis
- Etude de différentes approches
- Réalisation d'un prototype

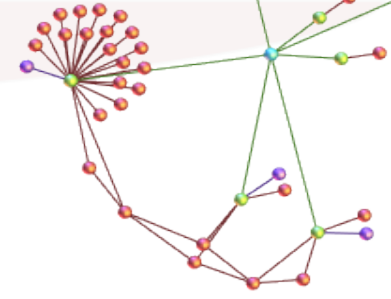


- **Principe de fonctionnement :**

- L'utilisateur peut zoomer et se déplacer sur l'axe
- On peut sélectionner une révision à étudier
- L'état du projet à la révision donnée est alors recalculé et affiché

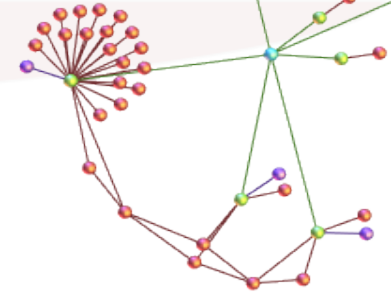
TX n°3497 CodeMiner





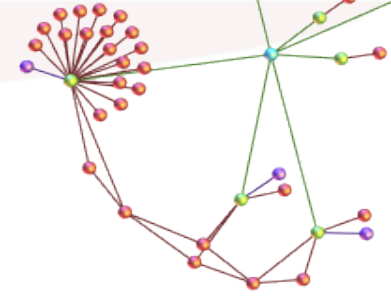
Difficultés surmontées

- **Plugin Eclipse**
 - Documentation de la librairie SVN
 - Gestion des événements et des Threads
- **Format de données**
 - Respect de la cohérence temporelle
- **Plugin Gephi**
 - Gestion de la concurrence lors de l'accès aux données
 - Interfaçage avec un logiciel en cours d'évolution



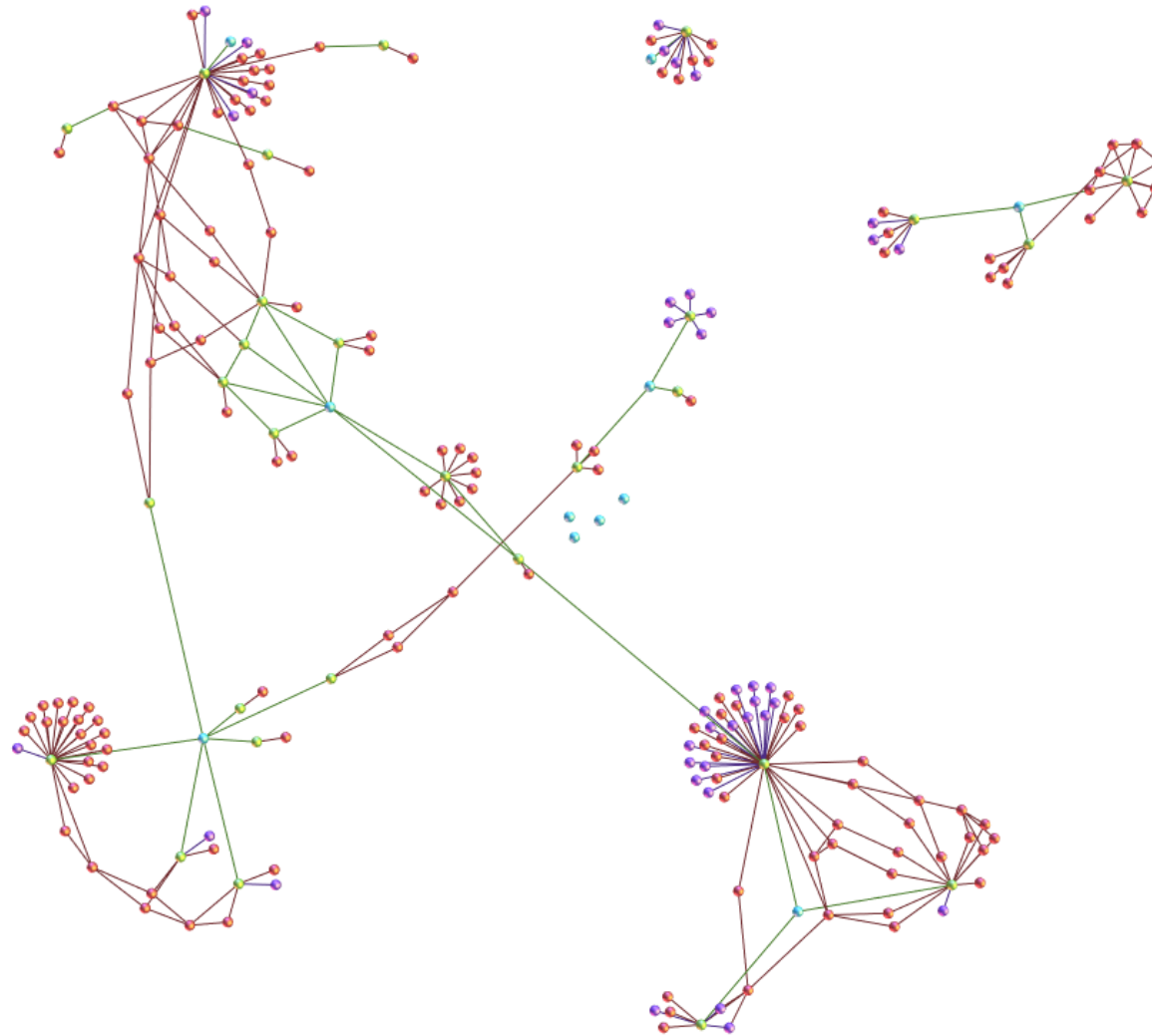
Bilan

- **Implémentation de la visualisation des dépôts SVN**
 - Appropriation des relations au sein d'un projet
 - Suivi de l'évolution de l'architecture au fil des révisions
- **Egalement des retombées positives sur Gephi**
 - Amélioration de la gestion du format de données
 - Création d'une interface pour des modules de navigation



Perspectives du projet

- **Perspectives à court terme**
 - Enrichissement de l'interface
 - Export de diagrammes de séquences UML
- **Perspectives au long terme**
 - Gestion de nouveaux langages
 - Ajout de types de dépôts (Bazaar, Git, Mercurial)
 - Production d'une solution de génie logiciel à part entière



Elèves-ingénieurs

Julian Bilcke

Sébastien Heymann

Microscope numérique de logiciels

"Network Dynamics"

Suiveur UTC

Mr Stéphane Crozat