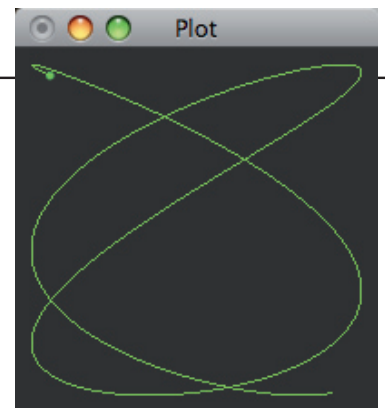


## en bref

*rubyk* est un logiciel open source utilisé pour effectuer du traitement de signal temps-réel. Il a en outre été utilisé pour transformer des données provenant de capteurs de mouvement (accéléromètres) en informations musicales (signaux midi).

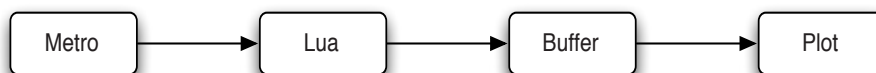
Voici un exemple de script rubyk qui affiche des figures de lissajous ainsi que le résultat dans l'objet **Plot**:

```
1 m = Metro(10000)
2 m => l
3 l = Lua("
4 i = 0
5 function bang()
6   i = i + 1
7   x1 = math.sin( i / 32)
8   y1 = math.sin( i / 50)
9   send(1, {x1,y1})
10 end
11 ")
12 l => b
13 b = Buffer(500)
14 b => p
15 p = Plot(mode:"xy" color:"fixed" red:0.4 green:1.0 blue:0.4 amplitude:1.1 width:200 height:200)
```



## fonctionnement

Le traitement du signal se fait par l'assemblage de «nœuds» connectés entre eux par des liaisons. Par exemple, le schéma fonctionnel du script ci-dessus est:



Le signal est transmis d'un nœud à l'autre chaque nœud effectuant une partie du traitement.

Par rapport à d'autres outils de ce type, *rubyk* se différencie par les aspects suivants:

- **source ouverte** (licence MIT)
- **extensibilité**: le logiciel a été pensé d'emblée pour faciliter la création de nouveaux types d'objets et/ou l'inclusion de bibliothèques mathématiques déjà écrites (une classe C++)
- forte intégration avec un langage de script (**Lua**) pour simplifier l'écriture de traitements simples
- code source testé (unit testing) et documenté (Doxygen)

Le logiciel fonctionne pour l'instant en ligne de commande mais une interface graphique devrait être disponible dans le courant 2009 suite au financement de l'office fédéral de la culture (sitemapping.ch).

## présentation de quelques nœuds

### Metro

C'est un des objets les plus simples: il envoie un «bang» à intervalles réguliers.

### Lua

Le signal entre par une des 10 entrées. L'entrée «1» déclenche la fonction *bang* qui doit être définie dans le script écrit en langage Lua. Si la définition du script est enregistrée dans un fichier externe, toute modification à ce fichier est automatiquement détectée et l'objet est mis à jour.

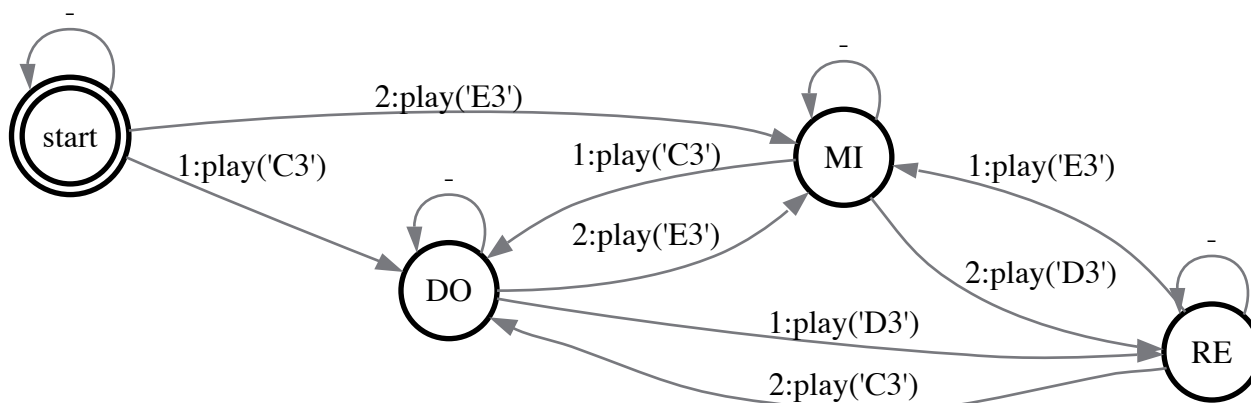
### Turing

L'objet Turing est une machine à état avec la possibilité de déclencher des fonctions en Lua lors des changements d'états. Le langage de définition de la machine à état a été conçu pour être très souple et rapide à écrire. Voici un exemple avec graphe de changements d'états correspondant (cet exemple joue les notes do, ré, mi en boucle en recevant le signal «1» et il joue mi, ré, do avec le signal «2» :

```

1  start 1:play('C3') -> DO 1:play('D3') -> RE 1:play('E3') -> MI 1:play('C3') -> DO
2  start 2:play('E3') -> MI 2:play('D3') -> RE 2:play('C3') -> DO 2:play('E3') -> MI
3
4  =begin lua
5  dofile("notes.lua")
6
7  function play(note)
8    send_note(1, N[note], 70, 500, 1, 0) -- port, note, velo, length, channel, time
9  end
10 =end
11
```

Graphe produit par l'outil *Graphviz* à partir du fichier «dot» généré par l'objet:



### PCA (Principal Component Analysis)

Objet permettant de réduire la dimensionnalité d'une matrice pour ne conserver que les composantes à forte variance. Cet objet a été optimisé pour le calcul parallèle en utilisant les bibliothèques BLAS et LAPACK.

### Svm

Classification de signaux en utilisant la technologie Support Vector Machine (utilisation de la bibliothèque de Chih-Chung Chang and Chih-Jen Lin).

### ClassRecorder

Outil facilitant l'enregistrement de séries de signaux pour différentes classes (affichage de la valeur moyenne, possibilité de recadrer un signal dans le temps, etc)

### MidiOut, Midiln

Entrées et sorties midi.

### Serial

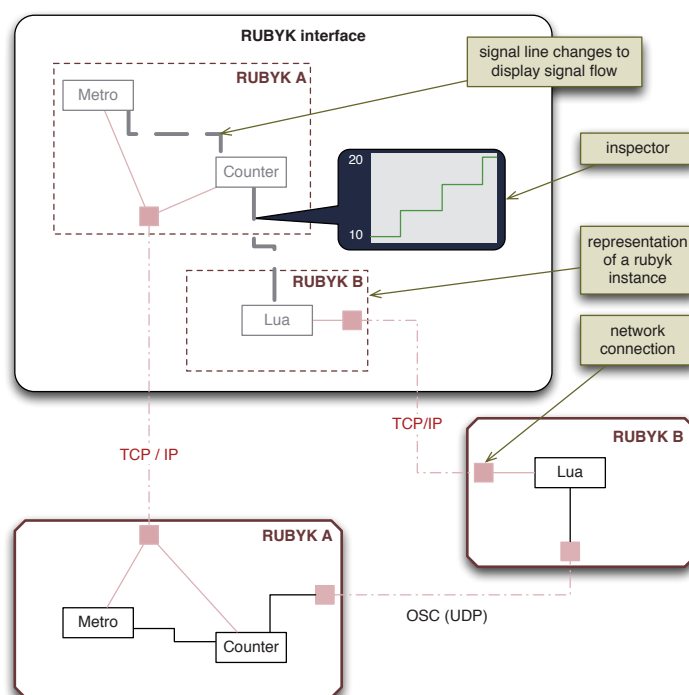
Entrées de données depuis le port série.

... etc

Et bien d'autres objets encore (Plot, Value, VQ, FFT, Buffer, ...). Voir [dev.rubyk.org/browser/trunk/objects](http://dev.rubyk.org/browser/trunk/objects).

## feuille de route

- septembre 08    définition d'un protocole de communication entre une interface et un serveur *rubyk*
- octobre 08    découverte de la librairie graphique Juce ([rawmaterialsoftware.com/juce](http://rawmaterialsoftware.com/juce)), implémentation d'une interface (sans lien avec rubyk). Travail sur le protocole interface-serveur de *rubyk*.
- nov. 08 - fév. 09    création d'une l'interface graphique basée sur Juce.
- mars 09 - juin 09    implémentation d'entrées/sorties vidéo
- juin 09 - ...    promotion de l'outil (site web, documentation)



projet d'interface graphique