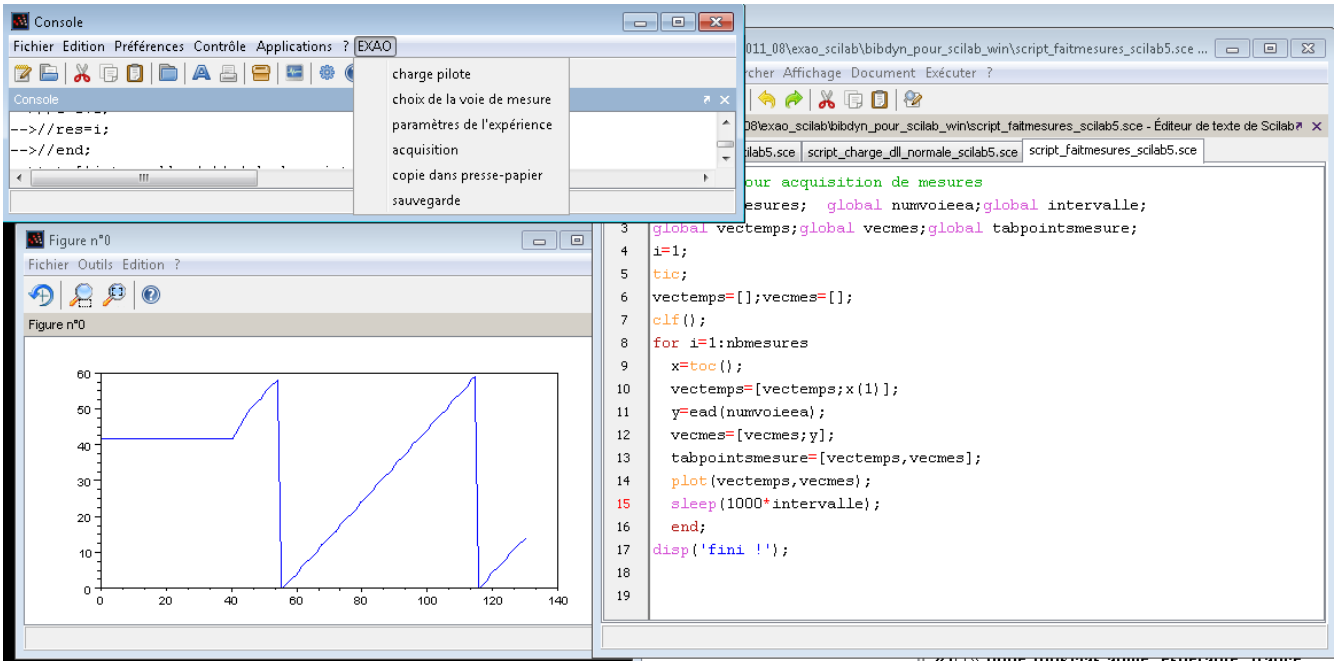
Pierre Dieumegard
professeur de SVT,
Lycée Pothier, 2 bis rue Marcel Proust,
F 45044 Orléans
courriel : pierre.dieumegard@ac-orleans-tours.fr

Mensurasoft-Scilab :

Mesures et expérimentations scientifiques en Scilab

Scilab est un logiciel de calcul numérique, très puissant pour analyser et visualiser les données, mais on peut aussi l'utiliser pour faire l'acquisition automatique des données, à l'aide d'un appareil de mesure connecté à l'ordinateur. Il existe pour divers systèmes d'exploitation, en particulier MS-Windows,



Linux, et Mac-OS. C'est un logiciel libre, développé initialement par l'INRIA, et actuellement par le consortium Scilab, membre de Digiteo ; on peut le télécharger sur http://www.scilab.org/.

Le système proposé ici repose sur l'emploi de bibliothèques dynamiques spécifiques des divers appareils de mesure (extensions classiques .so pour Linux et .dll pour MS-Windows). Ces petits fichiers jouent le rôle de "pilotes d'appareils de mesure " (= "drivers") : le programme rédigé en langage Scilab appelle les fonctions de ces bibliothèques dynamiques. Ainsi, potentiellement, ils rendent Scilab capable de converser avec tous les divers appareils de mesure.. C'est le système Mensurasoft, qui permet d'utiliser un grand nombre de langages informatiques (C++, Pascal, Basic, Scilab, Freemat, Logo, Python...), sous divers systèmes d'exploitation (MS-Windows, Linux...).

Les réalisations présentées ici ont été réalisées avec Scilab 5.2. On peut réaliser les mêmes fonctionnalités avec Scilab 4, mais certaines fonctions ont une syntaxe un peu différente, ce qui oblige à modifier un peu le détail de la programmation des dialogues. L'essentiel est conservé entre Scilab4 et Scilab 5 : l'appel des bibliothèques dynamiques.

1 Scilab peut utiliser les bibliothèques dynamiques, mais nécessite une syntaxe spéciale

Le système Mensurasoft est (relativement) simple, car il est destiné à permettre l'utilisation des pilotes

d'appareils de mesure par divers logiciels et langages informatiques, qui n'avaient pas été conçus spécialement pour lui. On peut trouver divers renseignements sur http://sciencexp.free.fr.

Les langages informatiques "classiques" permettent d'appeler (relativement) simplement les bibliothèques dynamiques ; par contre, Scilab nécessite des bibliothèques dynamiques spéciales, et la documentation est plutôt confuse.. Il serait bien sûr possible de réaliser des bibliothèques dynamiques spéciales de Scilab pour la communication avec les divers appareils de mesure, mais ces pilotes seraient inutilisables avec les autres logiciels et langages de programmation. C'est pourquoi la solution choisie ici a été de réaliser une bibliothèque dynamique d'adaptation de Scilab aux pilotes du système Mensurasoft (plusieurs dizaines d'appareils de mesure, développés depuis de nombreuses années).

1.1 Exigences particulières des bibliothèques dynamiques de Scilab

Les bibliothèques dynamiques de Scilab ont apparemment des exigences particulières :

- Elles ont un passage de paramètre de type "cdecl" (il existe plusieurs autres types, dont le plus fréquent est "stdcall"). Le système Mensurasoft est prévu pour les deux types.

- Elles ne permettent pas de renvoyer des chaînes de caractères, mais seulement des nombres. Ce n'est pas étonnant, puisque Scilab est fondamentalement un logiciel de calcul numérique, mais c'est gênant pour le système Mensurasoft, où la reconnaissance des voies de mesure est basée sur leur nom, sous forme d'une chaîne de caractères.

- Les paramètres numériques que l'on passe aux fonctions doivent être de type "pointeur d'entier", alors que le système Mensurasoft utilise des bibliothèques dynamiques dont les paramètres numériques sont des entiers.

Donc Scilab ne peut pas utiliser directement les bibliothèques dynamiques du système Mensurasoft.

1.2 Comment Scilab appelle ses bibliothèques dynamiques

La première opération est de faire la liaison entre le fichier de la bibliothèque et une fonction.

Par exemple, pour la fonction fondamentale de mesure, une entrée analogique qui renvoie une valeur réelle de type "double précision", on suppose que la bibliothèque dynamique contient une fonction nommée normead :

```
numh=link('pourscilab.dll','normead','c');
```

fait le lien de Scilab avec la fonction normead qui est contenue dans le fichier pourscilab.dll.

Ensuite, on peut déclarer la fonction Scilab :

```
function res=ead(n);
    res=call('normead',n,1,'i','out',[1,1],2,'d');
endfunction;
```

L'instruction call appelle la fonction normead de la bibliothèque dynamique, qui a comme paramètre d'entrée un pointeur d'entier ("i") et comme paramètre de sortie un réel de type double précision ("d").

1.3 Il faut un adaptateur entre Scilab et les pilotes d'appareils de mesure du système Mensurasoft

Après étude de la documentation (plutôt confuse) sur la programmation des bibliothèques dynamiques de Scilab, il apparaît qu'aucune fonction normale du système Mensurasoft ne convient.

D'une part pour les fonctions renvoyant des valeurs numériques, Scilab demande des paramètres de type pointeur d'entier, alors que Mensurasoft utilise directement des paramètres entiers, et il semble qu'il faille aussi un paramètre variable d'entrée. D'autre part, Scilab ne semble pas capable de renvoyer des chaînes de caractères, alors que Mensurasoft fonde la reconnaissance des voies de mesures sur le nom des voies, qui est une chaîne de caractères.

La solution est de faire une bibliothèque dynamique nommée par exemple pourscilab (.dll sous Windows, .so sous Linux), qui elle-même appellera une bibliothèque dynamique du système Mensurasoft. Lorsque Scilab appellera une fonction de pourscilab, cette fonction appellera la fonction correspondante du pilote Mensurasoft.

L'exemple proposé ici a été rédigé à l'aide du langage FreePascal. Il n'exporte que trois fonctions : chargedllnormale pour charger une bibliothèque dynamique normale (de Mensurasoft), normead pour faire la mesure sur une entrée analogique, normnead pour donner le nom de ces entrées analogiques.

```
library pourscilab_lazarus_mini;
uses sysutils, Dynlibs;
//pour l'adaptation des pilotes d'appareils
de mesure de Mensurasoft à Scilab
//à compiler par Lazarus avec l'option de
compatibilité Delphi
//Pour Delphi : remplacer Dynlibs par
wintypes
//simple programme de démonstration :
//pour la version complète, voir
pour_scilab_lazarus.pas
type pdouble=^double;
     pword=^word;
var nead:function(n:integer):pchar;stdcall;
var ead:function(n:integer):double;stdcall;

function chargedllnormale( ch:pchar;var
res:word):word; cdecl; export;
var L:thandle; chloc:string;
begin
chloc:=string(ch);
  res:=ord(chloc[1]);
  chargedllnormale:=res;
     @ead:=nil;
     @nead:=nil;
     L:=loadlibrary(ch);
     @nead:=getprocaddress(L,'stdnead');
     @ead:=getprocaddress(L,'stdead');
     end;

function normnead(n,i:pword;var
res:word):word;cdecl;export;
var chloc:string;   wloc:word;
begin
wloc:=n^;
chloc:=string(nead(wloc));
wloc:=i^;
if length(chloc)>0 then res:=ord(chloc[wloc])
else res:=0;
normnead:=res;
end;

function normead(n:pword;var
res:double):double;cdecl; export;
var wloc:word;
begin
 wloc:=n^;
 res:=ead(wloc);
normead:=res;
end;

exports
  chargedllnormale,normnead,normead;
begin
end.
```

Pour exploiter les diverses autres fonctions des pilotes du système Mensurasoft (sorties analogiques, entrées et sorties binaires...), il faut bien sûr d'autres fonctions.

1.4 Des fonctions Scilab pour communiquer avec cet adaptateur

Ce script scilab utilise l'adaptateur pourscilab_mini.dll. Il commence par demander à l'utilisateur le

nom du pilote à utiliser, puis il appelle l'adaptateur avec la fonction chargedllnormale, qui comme son nom l'indique, charge une bibliothèque dynamique du type Mensurasoft.

La fonction ead appelle simplement la fonction normead de l'adaptateur. La fonction nead est un peu plus compliquée, car elle reconstitue une chaîne de caractères complète (le nom de la fonction) à partir des différents caractères qu'elle appelle progressivement : lorsqu'elle reçoit un caractère nul, c'est que la chaîne est terminée.

Là encore, pour gérer les sorties analogiques et les entrées et sorties binaires, il faut d'autres fonctions.

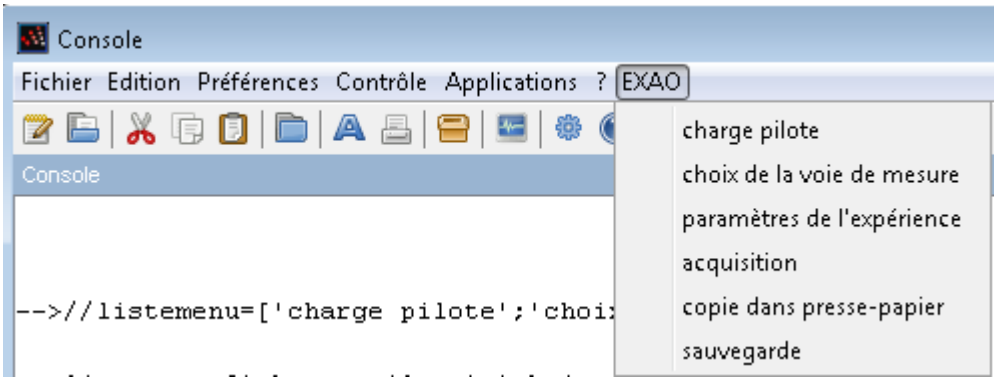
```
// Script pour pouvoir utiliser avec Scilab
les pilotes
// d'appareils de mesure ou d'actionneurs
d'expérimentation
// assistée par ordinateur.
// Consultez http://sciencex.free.fr
//Auteur : Pierre Dieumegard
(pierre.dieumegard@ac-orleans-tours.fr)
// Logiciel libre -- libera programaro --
free software --
global numh
if getos()=='Windows'
  dir('*.dll')
else
  dir('*.so')
end
nomadaptateur='./pourscilab_lazarus2.dll'; /
/à modifier selon les circonstance
nomfich=input('entrez le nom de la
bibliothèque à  utiliser','s');
if length(nomfich)>1
  numh=link(nomadaptateur,'chargedllnormale'
,'c');
  x=call('chargedllnormale',nomfich,1,'c','o
ut',[1,1],2,'i');
  numh=link(nomadaptateur,'normnead','c');
  numh=link(nomadaptateur,'normead','c');
 else disp('pas de pilote chargé !');
end;

function res = normnead(n,i);
    res=uint8(call('normnead',n,1,'i',i,2,'i
','out',[1,1],3,'i'));
endfunction;

function res=nead(n);
    ch='';
    i=1;
    while normnead(n,i)>0
        ch=ch+char(normnead(n,i));
        i=i+1;
    end;
    res=ch;
endfunction;

function res=ead(n);
    res=call('normead',n,1,'i','out',
[1,1],2,'d');
endfunction;

disp('vous pouvez maintenant utiliser les
fonctions suivantes :');
disp('ead et nead');
```

2 Réalisation d'un (mini)logiciel d'expérimentation en Scilab

2.1 Mise en place d'un menu supplémentaire de la console Scilab

Parmi la multitude de fonctions Scilab, certaines permettent d'ajouter des éléments de menu, ou d'ouvrir des boites de dialogue.

Pour réaliser un (mini)logiciel d'EXAO en Scilab, on peut installer un menu avec différentes fonctions :



- charger un pilote d'appareil de mesure : il faut commencer par ça.

- choisir la voie de mesure parmi celles qui sont disponibles dans l'appareil de mesure (cela peut varier de 1 pour un thermomètre, à plusieurs, par exemple 6 sur Arduino Uno, ou 8 pour Orphy Portable. C'est ici qu'il y a besoin de la fonction donnant le nom des voies de mesure : les voies réellement existantes sont celles qui ont un nom de longueur non nulle.

- choisir les paramètres de l'expérience : intervalle entre deux points de mesure, nombre de points de mesure dans l'enregistrement.

- faire l'acquisition elle-même, en visualisant dans une fenêtre graphique la courbe constituée par les différentes valeurs mesurées en fonction du temps.

```
// script qui ajoute un élément de menu EXAO à droite du menu scilab
global numvoieea ; numvoieea=0;
global intervalle ; intervalle=1;
global nbmesures ; nbmesures=100;
global vectemps;global vecmes;global tabpointsmesure;

function sauvedonnees(T)
  nomfich=uigetfile('*.tab');
  write_csv(T,nomfich);
endfunction

function copiedonnees(T)
  clipboard("copy",sci2exp(T));
endfunction

function res=action_choixvoie2;
global numvoieea;
if exists('detail')
  tabch=[];
  i=0;
  while length(nead(i))>0
    tabch=[tabch;nead(i)];
    i=i+1;
    end;
  res=x_choose(tabch,'choix de la voie ('+detail()+')','annule')-1;
  else
  disp('pas de pilote disponible !');
  res=-2;
  end;
  numvoieea=res;
endfunction;

function res=action_param_exp;
global numvoieea;global intervalle ; global nbmesures;
txt=['intervalle (s)';'nb de points de mesure'];
 sig=x_mdialog(['paramètres de l''expérience'],txt,[string(intervalle);string(nbmesures)])
 intervalle=evstr(sig(1));
 nbmesures=evstr(sig(2));
 res=0;
 endfunction;

listemenu=['charge pilote';'choix de la voie de mesure';'paramètres de
l''expérience';'acquisition';'copie dans presse-papier';'sauvegarde'];
addmenu('EXAO',listemenu) ;
EXAO=['exec(''script_charge_dll_normale_scilab5.sce'');';'action_choixvoie2;';'action_para
m_exp;';'exec(''script_faitmesures_scilab5.sce'');';'copiedonnees(tabpointsmesure)';'sauve
donnees(tabpointsmesure)'];
```

- ensuite, envoyer ces résultats de mesure vers d'autres logiciels par l'intermédiaire du presse-papier (pour Windows comme pour Linux), et/ou les enregistrer dans un fichier, par exemple un fichier-texte où les différents points de mesure correspondent aux différentes lignes.L'acquisition des données

C'est là que sera utilisée la fonction d'entrée analogique, sur la voie qui a été choisie dans la deuxième option du menu précédent.

2.2 Faire les mesures

```
//script pour acquisition de mesures
global nbmesures;  global numvoieea;
global intervalle;
global vectemps;global vecmes;global
tabpointsmesure;
i=1;
tic;
vectemps=[];vecmes=[];
clf();
for i=1:nbmesures
  x=toc();
  vectemps=[vectemps;x(1)];
  y=ead(numvoieea);
  vecmes=[vecmes;y];
  tabpointsmesure=[vectemps,vecmes];
  plot(vectemps,vecmes);
  sleep(1000*intervalle);
  end;
disp('mesures finies !');
```

3 Comment améliorer ce programme ?

Dans le script d'acquisition de données, on peut faire la moyenne de mesures rapprochées, plutôt que de prendre un point de mesure toutes les n secondes.

On pourrait aussi faire des mesures sur deux ou plusieurs voies, si l'appareil de mesure et le pilote le permettent.

En utilisant les autres fonctions de sorties analogiques, et d'entrées et sorties binaires, on pourrait faire une régulation d'une grandeur, ou ne déclencher les mesures que si une valeur dépasse un seuil.

Au lieu de faire des mesures toutes les n secondes, on pourrait faire des mesures toutes les n millisecondes, en faisant une sorte d'oscilloscope. Pour ceci, il ne faudrait pas retracer le graphique à chaque nouvelle mesure, mais seulement après la prise de l'ensemble des points de mesure d'une série.