Implantation d'un OS multi-processus pour téléphone cellulaire à l'aide d'automates

À travers ce projet vous découvrirez les bases du fonctionnement d'un OS (*Operating System* = système d'exploitation) multi-processus. Et vous apprendrez à utiliser des automates pour concevoir un système robuste.

1 Description du projet

L'objectif du projet est de réaliser un simulateur capable de faire fonctionner simultanément plusieurs automates et de l'utiliser pour concevoir une implantation robuste du mécanisme de gestion des différents applications d'un téléphone cellulaire et de leurs interactions. L'utilisation d'automates oblige à spécifier l'état de l'application pour chaque événement possible et évite les bugs du « baby player » où l'application entre dans un cas non spécifié lorsqu'un enfant actionne au hasard les éléments de l'interface.

On prend comme exemple le cas d'un téléphone portable mais on pourrait faire le même exercice sur le menu d'un site web ou toute autre interface.

On considère un téléphone avec les fonctionnalités suivantes disponibles sur le clavier :

1. réglage du volume : `+`,`-`
2. lecteur mp3 : `M` lance le « music player », `P` = (play,pause), `S` stop
3. appel : `0`,`1`,`2`,`3`,`4`,`5`,`6`,`7`,`8`,`9` numérotation, `E` effacer, `C` composer le numéro (c'est-à-dire appeler)
4. envoi de SMS : `N` rédiger un nouveau message, `0`,`1`,`2`,`3`,`4`,`5`,`6`,`7`,`8`,`9` rédaction (pour réduire l'alphabet on ne considère qu'un clavier numérique), `E` effacer, `V` envoyer
5. `A` annuler l'action précédente
6. `Q` quitter le menu actuel et revenir au précédent
7. prise d'appel (vous ou votre interlocuteur) : `D` décrocher, `R` raccrocher
8. reception de SMS : `L` Lire, `E` effacer
9. `O` on, `F` off

et les événements suivants venus de l'extérieur sont aussi considérés comme des symboles de l'alphabet de l'automate :

1. (`#`) sonnerie signalant la réception d'un appel
2. (`@`) sonnerie signalant la réception d'un message
3. (`D`) l'un des deux correspondants a décroché
4. (`R`) l'un des deux correspondants a raccroché

Ainsi, $\Sigma = \{$`+,-,M,P,S,0,1,2,3,4,5,6,7,8,9,E,C,N,V,A,Q,D,R,L,O,F,#,@,L,R`$\}$

Niveau de détails exigé dans le projet Dans cet exercice on ne s'intéresse pas aux fonctionnalités associées aux événements, ni au temps de réaction (pas de compteurs du temps écoulé) ; seuls nous importe le mode dans lequel se trouve le téléphone et la garantie qu'il ne se retrouve pas dans un mode dont on ne peut pas sortir, ni dans un mode non spécifié.

2 Principe de fonctionnement

On considére des automates dont les états sont des modes de fonctionnement et les transitions portent à la fois un événement déclencheur et une action.

Par exemple voici une transition de l'automate du « volume manager »

$$\boxed{\textsc{Normal}} \xrightarrow[\mathsf{Send}(\overrightarrow{\mathit{mute}})]{\mathit{decrease}} \boxed{\textsc{Mute}}$$

qui indique que dans l'état NORMAL si on fait décroitre le volume (événement *decrease*), on effectue l'action $\mathsf{Send}(\mathit{mute})$ qui génère et on envoie l'événement *mute* et on passe en mode MUTE.

Et voici une transition de l'automate du « music player » qui réagit à l'événement *mute* envoyé et passe dans l'état PAUSE si le volume est coupé. Notez que la transition ne génère pas de nouvelle action :

$$\boxed{\textsc{Play}} \xrightarrow[\emptyset]{\mathit{mute}} \boxed{\textsc{Pause}}$$

Enfin, voici une transition de l'OS qui lance l'automate « music player » (noté Aut_{mp}) lorsqu'on appuie sur la touche `M` qui provoque l'envoi de l'événement *music*.

$$\boxed{\textsc{Running}} \xrightarrow[\mathsf{Create}(\overrightarrow{Aut_{mp}})]{\mathit{music}} \boxed{\textsc{Running}}$$

2.1 Les applications

Chaque application est décrite par un automate qui définit les réactions aux touches du téléphone et les interactions avec les autres applications.

Dans notre simulation de téléphone cellulaire, il y a juste une application particulière qui est supposée déjà créée même lorsque le téléphone est éteint et qui réagit à la touche `0`, c'est-à-dire l'événement *on*.

Les actions On considère trois types d'actions associées à une transition :
- aucune action,
- un envoi d'événement,
- une création d'automate

Ce qui se décrit facilement par un type OCAML `action`

Les événements sont

- soit générés par une action Send
- soit provoqués par l'appuie sur une touche du téléphone

Notez qu'**une même touche peut générer plusieurs événements**. Ainsi il est très courant sur un téléphone cellulaire que la même touche, disons [P], qui gère les événements *play* et *pause* (et c'est très pratique). C'est à l'application de savoir comment interpréter ces événements en fonction de son mode courant. Lorsqu'on appuie sur [P], elle génére les deux événements *play* et *pause*. Si le player est en mode PLAY, il réagit à l'événement *pause* et ignore l'événement *play*.

Les états des automates correspondent aux différents modes du téléphone qui comporte au minimum les modes suivants :

1. off,
2. on,
3. mp3,
4. numérotation,
5. appel,
6. rédaction sms,
7. envoi sms,
8. sonnerie tél,
9. sonnerie sms,
10. communication,
11. lecture sms,
12. ...

Le système complet est constitué de tous les processus actif. Le comportement de chaque processus est défini par un automate. Il doit être possible de créer plusieurs instances d'un même automate.

2.2 Fonctionnement du simulateur d'automates

Votre implantation devra prendre en entrée un scenario sous la forme d'une séquence de caractères consitutée de symboles de l'alphabet du système (à la fois les touches du clavier et les événements extérieurs). Votre moteur de simulation devra se comporter de la manière suivante :

1. afficher le mode/état courant (le nom du mode/état doit etre explicite) de chacun des processus actif,
2. lire/afficher l'événement courant
3. faire réagir chacun des processus à l'événement courant :

(a) déterminer les transitions possibles

(b) exécuter les actions correspondantes (création de processus et envoi d'événements)

(c) mettre à jour l'état de chacun des processus

2.3 Visualisation

Pour suivre l'évolution du système vous devez être en mesure d'afficher à chaque transition du système :

- les processus actifs
- l'état courant de chaque processus
- les événements courants

Si vous utilisez l'interprète OCAML il n'est pas utile de réaliser une fonction d'affichage pour suivre l'état du système. Il suffit de demander à l'interprète la valeur de la variable qui représente l'état du système.

Exemple de présentation

```
system =
{input =
  [(Key 'm', [OS_event Music]);
   (Key 'p', [Mp3_event Play; Mp3_event Pause]);
   (Key 'p', [Mp3_event Play; Mp3_event Pause]);
   (Key '+', [Vol_event Inc]);
   (Key '-', [Vol_event Dec]);
   (Key '-', [Vol_event Dec]);
   (Key '+', [Vol_event Inc]);
   (Key 'p', [Mp3_event Play; Mp3_event Pause]);
   (Key 's', [Mp3_event Stop]);
   (Key 'f', [Phone_event Off])
  ];
 processes =
  [{aut = "mp3 player"; id = 5; current_state = Mp3_state PLAY};
   {aut = "mp3 player"; id = 4; current_state = Mp3_state PAUSE};
   {aut = "volume manager"; id = 3; current_state = Vol_state NORMAL}
   {aut = "os"; id = 2; current_state = OS_state RUNNING};
   {aut = "phone"; id = 1; current_state = Phone_state ON};
  ]}
```

La variable `system` est ici un enregistrement à deux champs :

- `input` montre le scenario avec les événements associés à chaque touche
- `processes` montre l'état des processus actifs

Conseil Pour tester votre automate il est plus simple de faire une version qui lit des événements/caractères entrés un par un au clavier ; il ne sera pas diffile de l'étendre pour obtenir une implantation qui prend en entréé une chaine de caractère et qui est plus pratique pour une démo.

3 Consignes et évaluation du projet

Plus votre implantation sera claire, lisible et puissante, meilleure sera votre note.

On appréciera que votre implantation soit capable de gérer correctement plusieurs instances d'un même automate. En particulier plusieurs « communications » à la fois (avec mise en attente ou mode réunion), plusieurs « music player » ouverts simultanément, *etc.*

Soyez à la fois créatifs (proposez un téléphone innovant) et réalistes (ne compliquez pas inutilement le moteur du simulateur). Et soignez votre codage.

3.1 À rendre

- Un document de quelques pages décrivant les spécificités de votre téléphone, les modes et les symboles utilisés
- une implantation en OCAML
- un jeu de tests qui permet d'atteindre chacun des modes et de revenir au mode d'accueil.

3.2 Soutenance de 10min + 5 min de questions

La soutenance aura lieu lors du dernier TD. Attention 10 min c'est très court !

- 5 min de présentation des choix d'implantation
- 5 min de démo sur des scénarios préparés à l'avance qui montrent les fonctionnalités intéressantes de votre automate
- 5 min de questions techniques