UBO

Université de Bretagne Occidentale

DEPARTEMENT INFORMATIQUE

Travail d'étude et de recherche

Implantation de l'algorithme LLREF dans cheddar

Elaboré par :

[Salem Anis FADHLOUN]

Encadré par :

[M. Frank Singhoff]

Soutenu le 10/06/2016 devant le jury :

M.	Erwan FABIANI
M.	Frank SINGHOFF
Mme	Sophie GIRE
M.	François MONIN

Remerciements

Au terme de ce travail d'étude et de recherche, je tiens à adresser mes plus vifs remerciements à toutes les personnes qui, de près ou de loin, ont contribué à l'aboutissement de ce travail dans les meilleures conditions.

Je m'adresse en premier lieu aux membres du jury que je remercie d'avoir accepté d'évaluer ce travail.

Je remercie vivement mon encadrant **M. Frank Singhoff** qui s'est montré très coopératif et qui a su me consacrer son temps quand j'en avais besoin. Je le remercie énormément.

Je ne laisserai pas cette occasion passer sans exprimer ma reconnaissance envers tous mes enseignants ainsi qu'au personnel de l'Université de Bretagne Occidentale.

Sommaire

La liste des figures

Introduction générale

Ce projet s'inscrit dans le cadre d'un travail d'étude et de recherche pour valider la première année du master informatique spécialité Système Informatique Complexe à l'Université de Bretagne occidentale de Brest, France.
Nous avons réalisé notre stage au sein du laboratoire informatique Micro 2.4 et Micro 1.3. Le sujet est intitulé "Implantation d'un algorithme d'ordonnancement multi-cœurs dans Cheddar".

Notre projet consiste à étendre les capacités d'analyse de Cheddar pour les architectures multi-cœurs en implantant l'algorithme « **L**argest **L**ocal **R**emaining **E**xecution Time **F**irst ».

Ce rapport comporte les différentes étapes et méthodologies suivies pour la réalisation du projet. Il contient trois chapitres organisés comme suit : Le premier chapitre intitulé « Contexte Générale » est consacré à la présentation du contexte du travail ainsi que l'organisme d'accueil.

Le deuxième chapitre intitulé « Etudes théoriques » dont lequel nous faisons l'analyse de l'algorithme LLREF et nous y présenterons deux jeux de tâches pour mieux comprendre son fonctionnement.
Le troisième chapitre intitulé «Réalisation» présente l'environnement de travail ainsi que les outils logiciels que nous avons utilisés pour la réalisation de notre projet et il illustre le travail réalisé à travers un ensemble de capture d'écran de l'application en exécution.

En conclusion, nous mentionnons les différents atouts de ce projet et les perspectives d'améliorations possibles.

Chapitre I : Contexte Général

Introduction

Ce chapitre est consacré pour la présentation de l'organisme d'accueil, la précision du cadre du travail et la problématique puis l'énumération des étapes de travail à réaliser pour achever ce projet.

1. Présentation de l'organisme d'accueil

Université de Bretagne Occidentale

L'**U**niversité de **B**retagne **O**ccidentale est un établissement public à caractère scientifique, culturel et professionnel, l'UBO a pour missions premières de concourir au développement de la recherche et à l'élévation du niveau scientifique, culturel et professionnel de la nation et des individus qui la composent, à la croissance régionale et nationale, à l'essor économique et à la réalisation d'une politique de l'emploi. Elle concourt enfin à la réduction des inégalités sociales et culturelles.

L'université assure :

- La formation initiale et continue
- La recherche scientifique et technique ainsi que la valorisation de ses résultats
- La diffusion de la culture et l'information scientifique et technique
- La coopération internationale. [1]

2. Présentation du projet

2.1. Contexte

- Dans les systèmes d'exploitation, l'ordonnanceur désigne le composant du noyau du système d'exploitation choisissant l'ordre d'exécution des processus sur les processeurs d'un ordinateur.

- Le rôle de l'ordonnanceur est de permettre à tous ces processus de s'exécuter à un moment ou un autre et d'utiliser au mieux le processeur pour l'utilisateur. Pour que chaque tâche s'exécute sans se préoccuper des autres et/ou aussi pour exécuter les tâches selon les contraintes imposées au système : contraintes temporelles, ressources partagées etc.
- Le projet Cheddar a été lancé par l'équipe lab-STICC de l'université de Brest depuis 2002, il s'agit d'un outil qui permet de calculer et présenter l'ordonnancement des tâches selon différents algorithmes d'ordonnancement en temps réel. Ainsi il permet de modéliser des architectures logicielles des systèmes en temps réel et de vérifier ses critères de performance ordonnançabilité etc.

2.2. Objectifs

La version actuelle de Cheddar comporte un petit nombre d'algorithmes d'ordonnancement multi-cœurs.

L'objectif de ce TER consiste à étendre les capacités d'analyse de Cheddar pour les architecture multi-cœurs en implantant l'algorithme LLREF.

3. Démarches du travail

Le projet comprend trois phases :

- L'analyse de l'algorithme LLREF : il s'agit ici de comprendre son fonctionnement.
- Proposition d'un jeu de test pour cet algorithme d'ordonnancement.
- Programmation de l'algorithme au sein de Cheddar

Conclusion

Dans ce chapitre nous avons présenté l'organisme d'accueil et le projet à réaliser. Une étude théorique des besoins nécessaires fera l'objet du chapitre suivant.

Chapitre II : Etude de l'algorithme

Introduction

Dans ce chapitre nous allons analyser l'algorithme LLREF et définir deux jeux de tâches qui vont nous permettre de bien le tester. En effet ce chapitre va nous permettre de mieux comprendre le travail.

1. Analyse de l'algorithme

1.1 L'ordonnancement en T-L planes :

L'ordonnancement en T-L planes repose sur le principe qu'un bon ordonnancement pour un seul T-L Plane peut se répéter pour toutes les T-L planes vu que les T-L planes se répètent successivement au cours du temps.

Nous voulons dire par bon ordonnancement que nous serons capable de construire un ordonnancement de tous les tâches qui s'exécute dans le T-L plane qui est proche le plus possible du modèle d'ordonnancement fluide.

Dans un T-L Plane chaque tâche est représentée par un jeton. L'emplacement du jeton représente le temps courant sur l'axe des abscisses et le temps d'exécution restant sur l'axe des ordonnées.

Nous pouvons déterminer le statut d'une tâche (en exécution ou inactive) à travers le sens de déplacement du jeton qui lui correspond :

- Si un jeton se déplace horizontalement, ceci signifie que la tâche correspondante est inactive.
- Si un jeton de déplace diagonalement ça signifie que la tâche correspondante est en cours d'exécution.

La figure 1 présente un exemple d'un T-L plane :

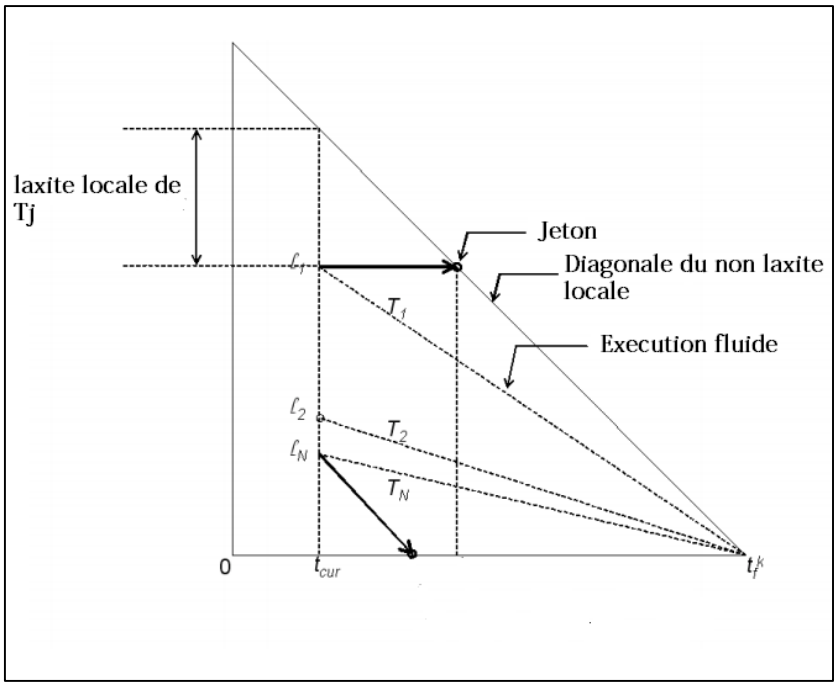

Figure 1 : Schéma d'un ordonnancement en T-L plane

1.2 L'algorithme LLREF (Largest Local Remaining Execution time First)

L'algorithme LLREF consiste à ordonnancer les tâches dans les limites d'un T-L plane.

LLREF choisit les tâches (le nombre des tâches à choisir dépend du nombre des processeurs) qui ont le temps d'exécution restant le plus grand.

En tout instant d'ordonnancement :

- Les m (au plus) jetons associés aux plus grandes durées d'exécution locale restantes non nulles adoptent un déplacement en diagonale, ce que signifie les m (au plus) tâches associées aux plus grandes durées d'exécution locale restantes non nulles s'exécutent.

- Les autres jetons se déplacent horizontalement, ce que signifie que les autres tâches sont suspendues.

-La limite d'utilisation de LLREF est : $\sum Ui \leq m$ avec $Ui = Ci / Pi$ et m : le nombre de processeurs disponibles.

- Si une tâche a été élue par l'ordonnanceur, elle commence son exécution et elle ne peut être suspendue que si l'un des deux événements suivant se produit :

- ✓ **Evénement B :** (Bottom hitting event) : cet événement se produit lorsqu'un jeton croise l'axe des abscisses, ce qui signifie que la tâche a terminé son exécution et par suite l'ordonnanceur doit choisir une autre tâche pour s'exécuter.

- ✓ **Evénement C:** (Ceilling hitting event): cet évènement se produit lorsqu'un jeton croise la ligne diagonale: ceci signifie que la tâche en question doit s'exécuter immédiatement pour qu'elle ne rate pas son échéance. [1]

La figure 2 illustre le fonctionnement des deux évènements C et B :

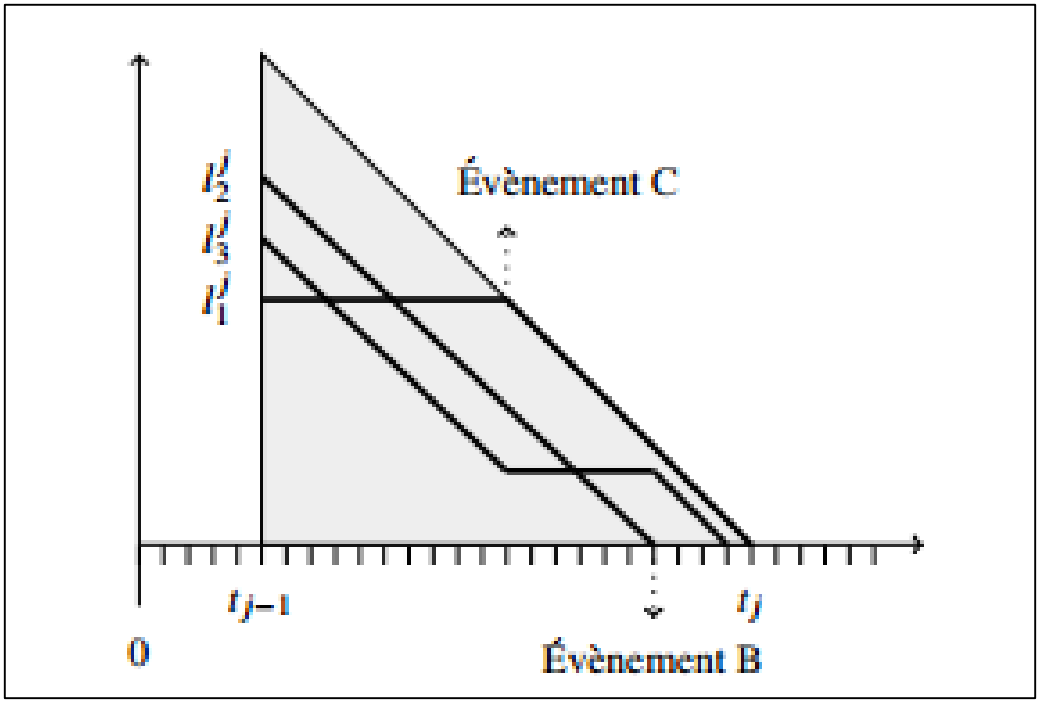

Figure 2: schéma des événements B et C

2. Exemples :

2.1 Premier jeu de tâches :

Prenons le jeu de tâches suivant que nous allons ordonnancer sur 2 processeurs :

Nom de la Tâche	T1	T2	T3
Capacité	5	7	9
Période	11	11	11
Date limite	11	11	11

2.1.1 Représentation en T-L Plane :

La figure 3 l'ordonnancement de notre jeu de tâches en T-L plane :

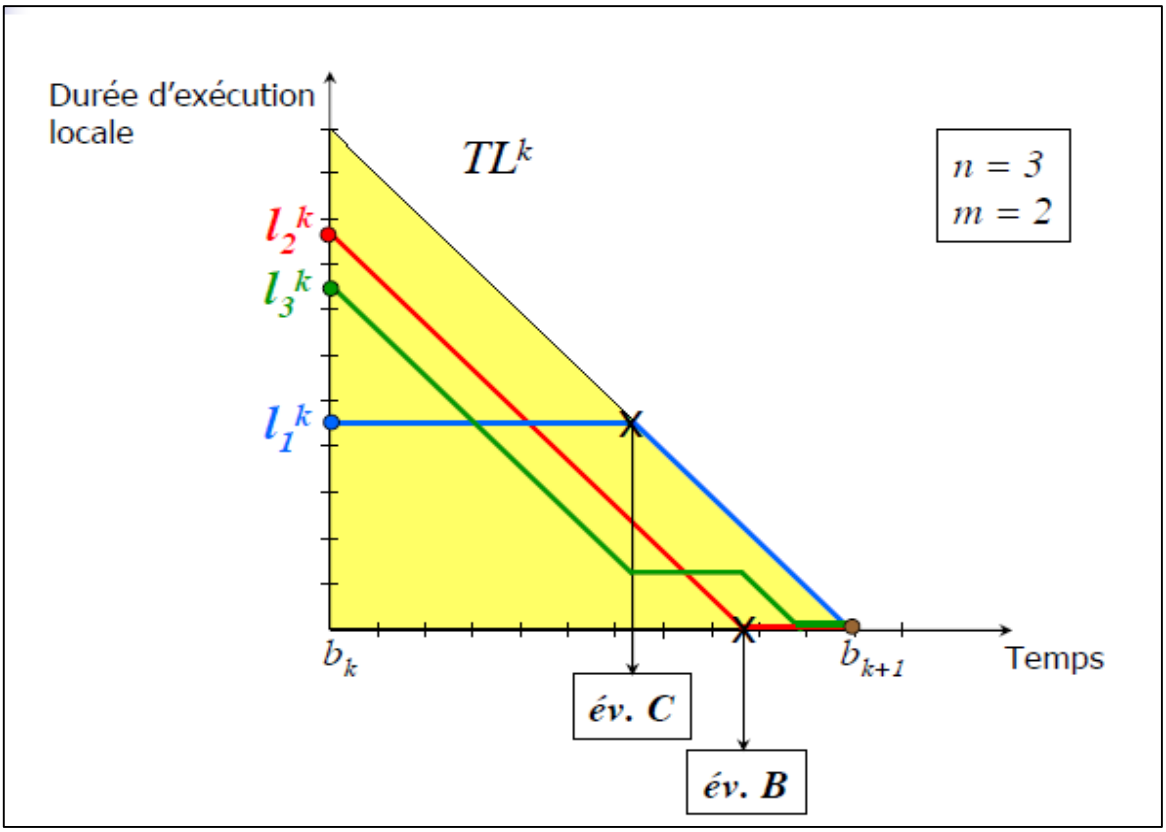

Figure 3: Ordonnancement du premier jeu de tâches en T-L plane

2.1.2 Représentation en chronogramme :

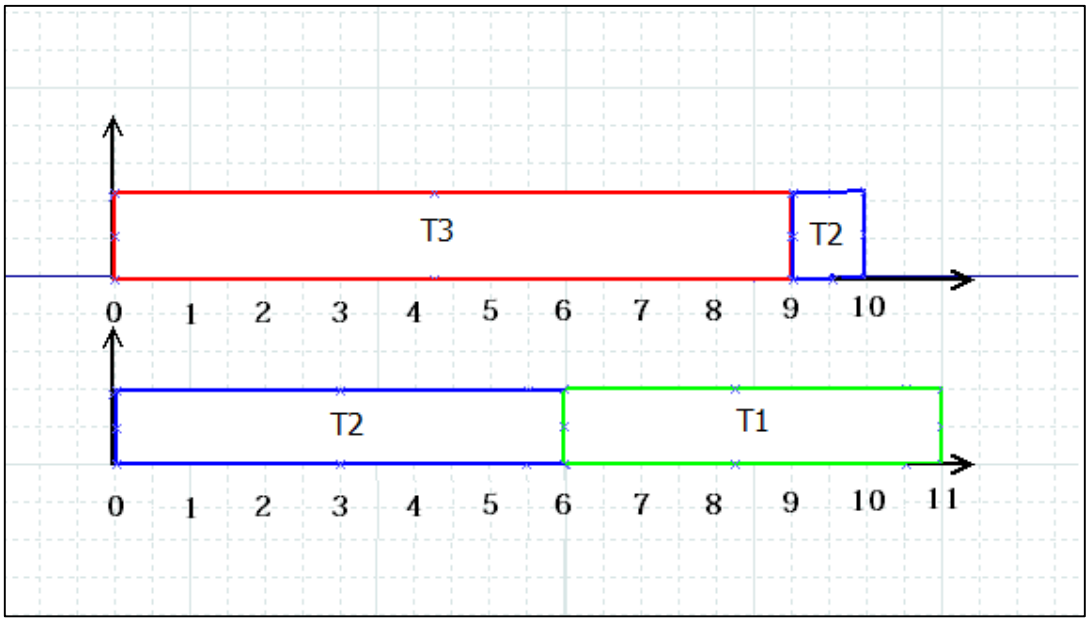

Figure 4: Ordonnancement du premier jeu de tâches en chronogramme

2.2 Deuxième jeu de tâches :

Soit le jeu de tâches suivant que nous allons ordonnancer sur 2 processeurs aussi :

Nom de la Tâche	T1	T2	T3	T4
Capacité	7	6	5	1
Période	10	10	10	10
Date limite	10	10	10	10

2.2.1 Représentation en T-L Plane :

La figure 5 montre l'ordonnancement de notre jeu de tâches en T-L plane :

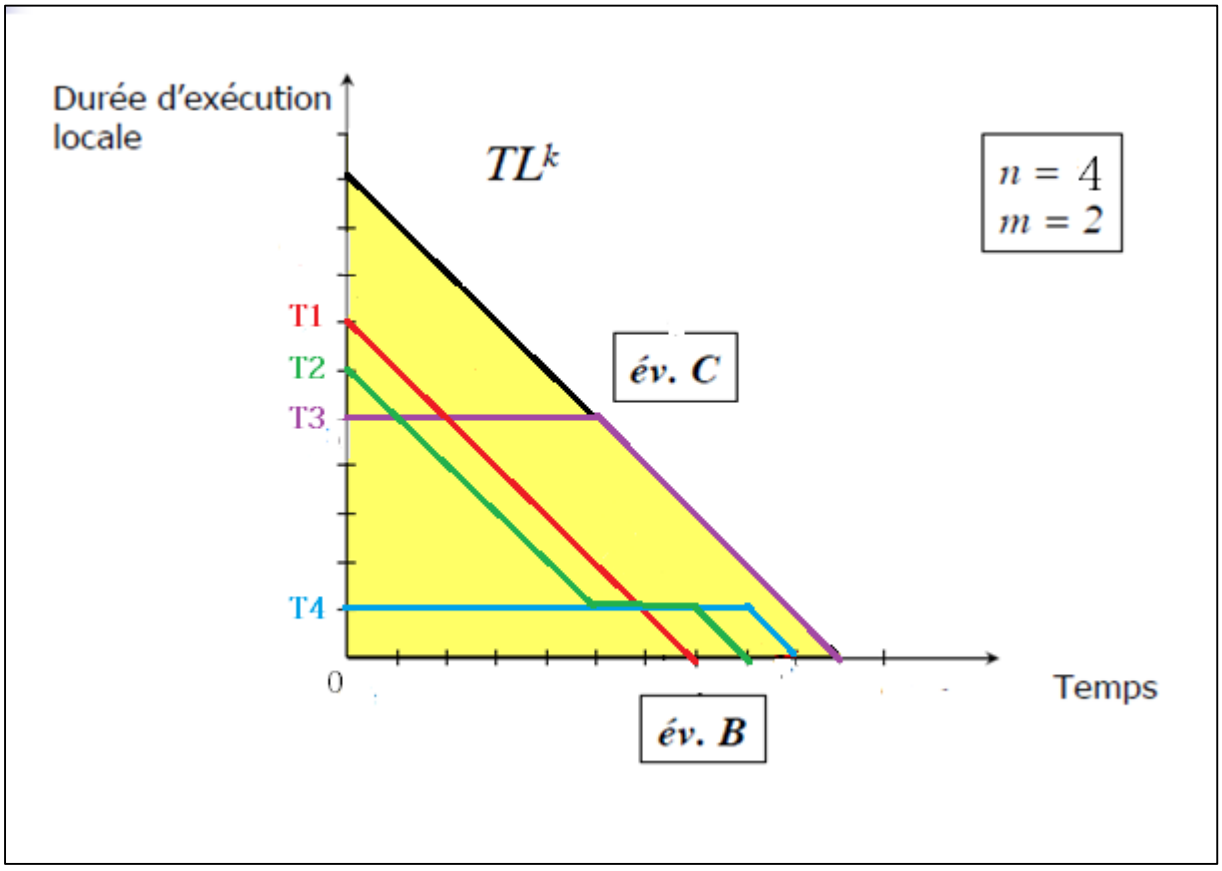

Figure 5: Ordonnancement du deuxième jeu de tâches en T-L plane

2.2.2 Représentation en chronogramme :

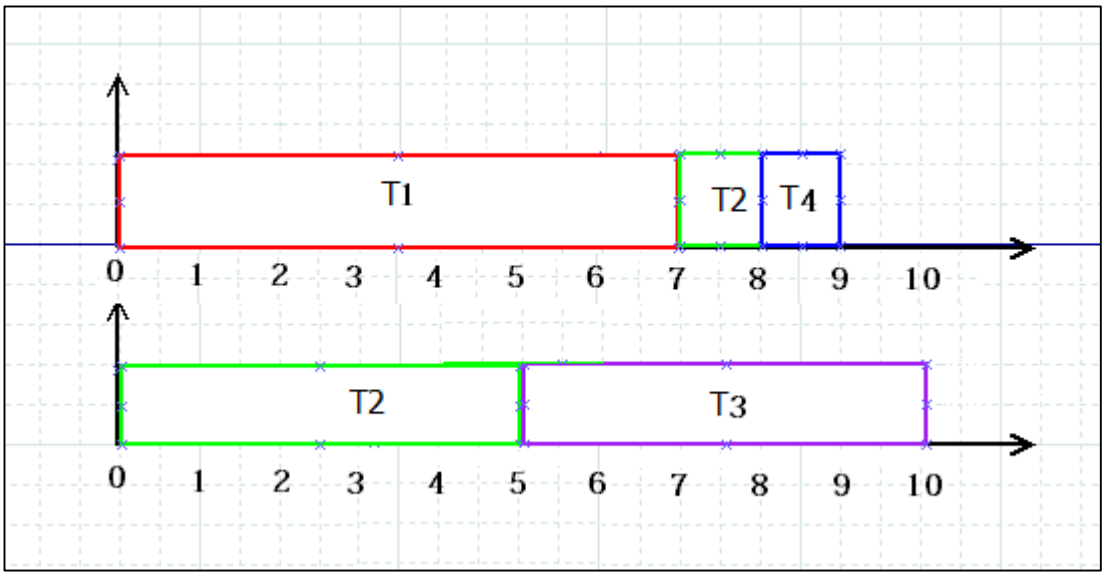

Figure 6: Ordonnancement du deuxième jeu de tâches en chronogramme

Conclusion

Dans ce chapitre nous avons pu analyser l'algorithme LLREF afin de comprendre son fonctionnement et nous avons réussi à l'appliquer sur deux exemples différents de jeu de tâches. Dans le chapitre suivant nous allons entamer la phase d'implémentation dont laquelle nous appliquerons ce qu'on a appris sur LLREF afin de l'implémenter sur Cheddar.

Chapitre III : Réalisation

Introduction

Une des étapes du cycle de vie d'un projet est aussi importante que l'étude théorique, est l'implémentation. Cette étape constitue la phase d'achèvement et d'aboutissement du projet. Pour accomplir cette tâche avec succès il faut savoir utiliser les outils techniques adéquats et nécessaires. Ce choix d'outils peut influencer sur la qualité du produit obtenu et donc nécessite une attention particulière et doit se baser sur les besoins du projet et le résultat escompté.

1. Environnement de travail

1.1. Environnement matériel

→ Ordinateur Portable HP pavillon g6 avec :

- Processeur Intel® core i5
- RAM : 6 Mo
- Stockage interne : 1024 Go
- Ecran 15,6 pouces
- Système d'exploitation : Windows 8.1

1.2. Environnement Logiciel

1.2.1 Cheddar

Cheddar est un outil de simulation conçu à l'Université de Bretagne Occidentale, il est développé en Ada95 et devrait fonctionner sur toutes les plates-formes supportées par GNAT. Cheddar permet de calculer différents critères de performance (contraintes temporelles, dimensionnement de ressources). L'outil permet, entre autre, de tester le respect des contraintes temporelles d'un jeu de tâches modélisant une application/un système temps réel. Cet outil a initialement été écrit dans un but pédagogique, mais il peut néanmoins, dans une certaine mesure, aider les personnes qui conçoivent des algorithmes d'ordonnancement temps réel ou qui souhaitent étudier les contraintes temporelles d'une application temps réel simple [SIN].
Cheddar est principalement constitué de deux composants logiciels :

- Un éditeur qui permet à l'utilisateur de décrire l'application à analyser et sur lequel, les résultats de simulation seront présentés.

- Une bibliothèque comportant les principaux résultats de la théorie de l'ordonnancement temps réel ainsi que quelques outils de files d'attente.

Cheddar offre deux types de fonctionnalité : un environnement de simulation et des tests de faisabilité. Les tests de faisabilité permettent à l'utilisateur de vérifier si les contraintes temporelles d'une application sont respectées sans pour autant calculer une simulation de l'ordonnancement. L'environnement de simulation quant à lui, effectue une analyse d'un ordonnancement précédemment calculé. L'environnement peut être étendu afin de modéliser des modèles de tâches ou d'ordonnanceurs non couverts par la théorie de l'ordonnancement actuel.

Cheddar a plusieurs fonctionnalités, entre autres :

- Calcul d'ordonnancement pour la plupart des algorithmes "standards" : RM, EDF, DM, LLF…
- Possibilité d'appliquer des tests de faisabilité dans le cas préemptif ou non préemptif :

- Support des tâches périodiques, apériodiques.

- Expression et ordonnancement de tâches avec contraintes de précédence.
- Etc… [3]

La figure 7 représente l'ordonnancement d'un jeu de tâches avec Cheddar en utilisant l'algorithme « Highest Priority First ».

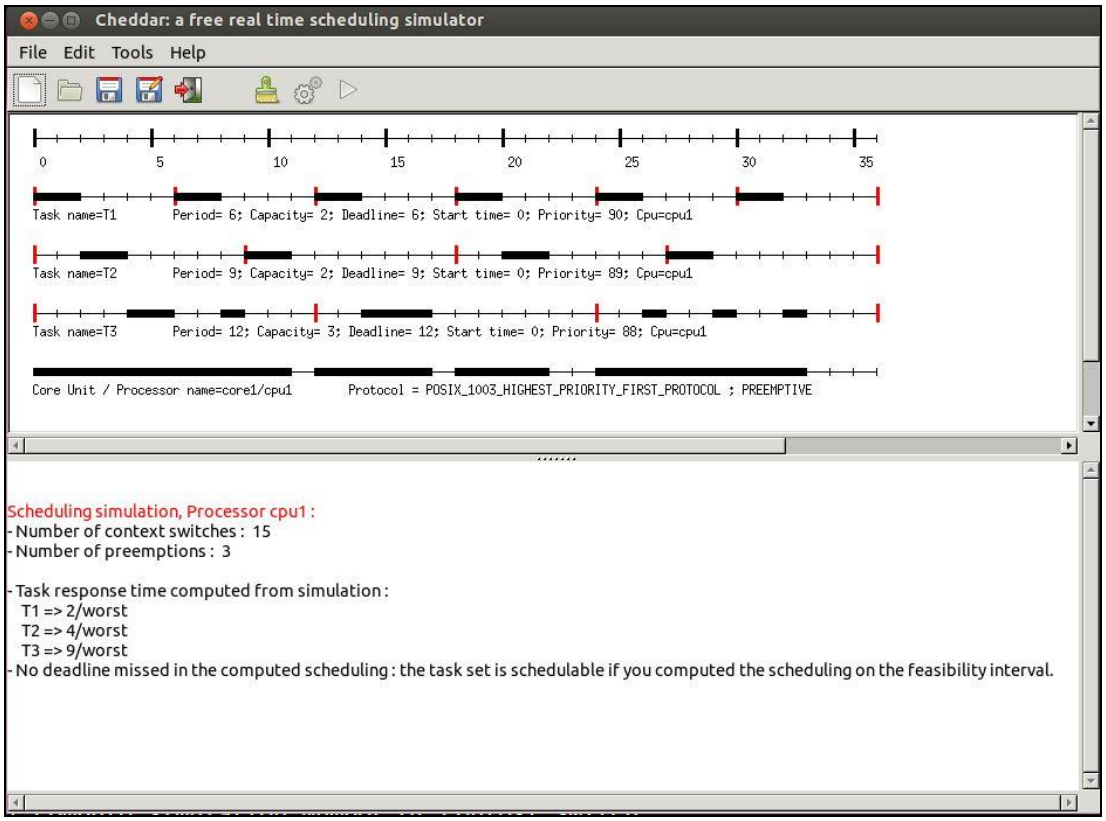

Figure 7: Exemple d'ordonnancement d'un jeu de tâches avec Cheddar

1.2.2 Virtual Box :

Virtual Box ou machine virtuelle est un logiciel virtualisation de systèmes d'exploitation. En utilisant les ressources matérielles de l'ordinateur *(*système hôte*)*, Virtual Box permet la création d'un ou de plusieurs ordinateurs virtuels dans lesquels s'installent d'autres systèmes d'exploitation. [4]

1.2.3 Langage de programmation ADA :

Ada est un langage de programmation qui a été normalisé en 1983 après avoir été conçu en réponse à un cahier des charges conçu par le DOD (département de la Défense) des États-Unis.

Ada est un langage à typage statique, modulaire et offrant une syntaxe claire inspirée de Pascal. Il est souvent utilisé dans des systèmes temps-réel nécessitant un haut niveau de fiabilité. [5]

1.2.4 GNAT 2012 :

GNAT est un compilateur logiciel libre pour le langage de programmation Ada, qui fait partie de la GNU Compiler Collection. Il prend en charge toutes les versions de la langue, à savoir Ada 2012, Ada 2005, Ada 95 et Ada 83. [6]

1.2.5 SVN :

Subversion (en abrégé svn) est un logiciel de gestion de versions, distribué sous licence Apache et BSD. Il a été conçu pour remplacer CVS. Ses auteurs s'appuient volontairement sur les mêmes concepts (notamment sur le principe du dépôt centralisé et unique). [7]

2. Mise en œuvre

Pour réaliser le travail demandé, nous avons implémenté le programme demandé dans deux fichiers texte ayant les extensions « .ads » (pour le fichier de spécification) et « .adb » (pour le fichier qui contient le corps du programme).

Vu la grande ressemblance de l'algorithme LLREF avec l'algorithme EDZL (Earliest DeadLine Zero Laxity) nous avons utilisé des morceaux de codes de ce dernier afin de nous faciliter la tâche en ajoutant les tests qui sont spécifiques et qui permettent de faire l'exécution selon l'algorithme LLREF.

2.1 Implémentation de l'algorithme en ada

LLREF a trois conditions qui le caractérisent :

La première condition est lorsqu'un évènement C se produit dans ce cas on doit exécuter immédiatement la tâche correspondante :

```
if  (Si.Tcbs (I).Rest_Of_Capacity = llref_Tcb_Ptr (Si.Tcbs (I)).tsk.Deadline - current_time)   then
                    put_line("    Evenement c ");
                       Elected := I;
                       No_Task := False;
                     return;
 end if;
```

La deuxième condition c'est de chercher la tâche qui possède le temps d'exécution restant le plus long : cette tâche sera élue par le processeur pour s'exécuter en premier.

```
if (Si.Tcbs (I).Rest_Of_Capacity > biggest_remaining_execution_time) then
                        biggest_remaining_execution_time := Si.Tcbs (I).Rest_Of_Capacity;
                        Elected           := I;
                        No_Task := False;
 end if;
```

La troisième condition est lorsque une tâche a commencé son exécution et n'a pas encore terminé sa capacité, dans ce cas la tâche doit continuer son exécution jusqu'à la fin de sa capacité.

```
if((Si.Tcbs (my_scheduler.previously_elected).Rest_Of_Capacity > 0) and (current_time /= 0) )  then
          Elected:=my_scheduler.previously_elected;
          No_Task := False;
          return;
end if;
```

2.2 Scénario d'exécution

La figure 8 représente l'interface d'accueil de Cheddar.

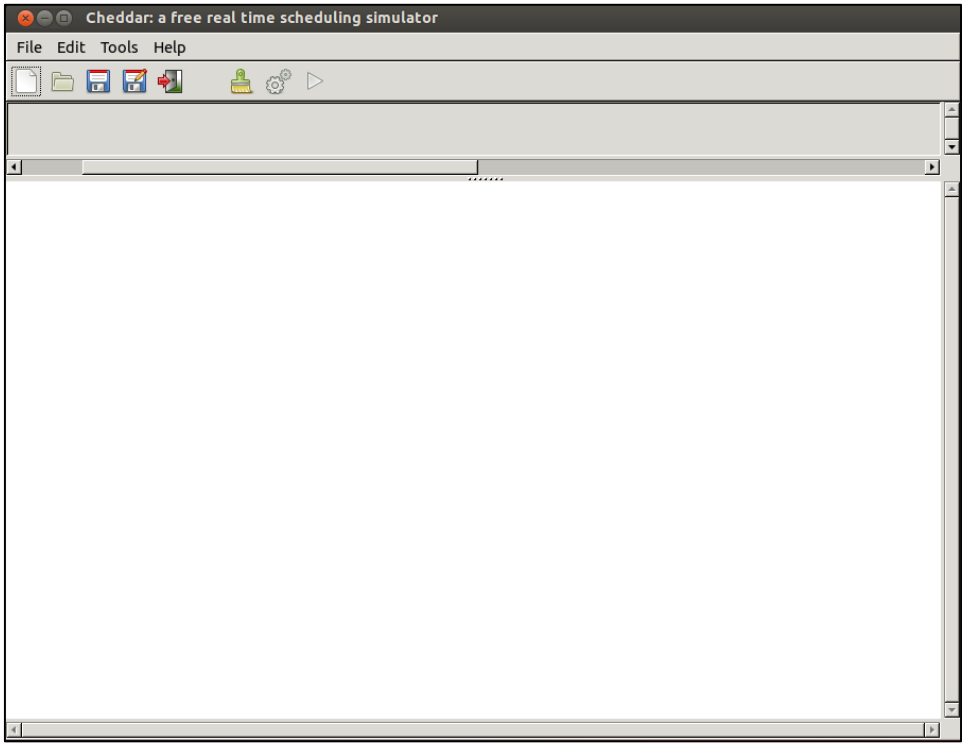

Figure 8: Interface d'accueil de Cheddar

Avant de faire une simulation pour un jeu de tâches il faut définir les cores qu'on va utiliser. La figure 9 montre que nous avons déclaré deux cores qui ordonnancement selon l'algorithme LLREF.

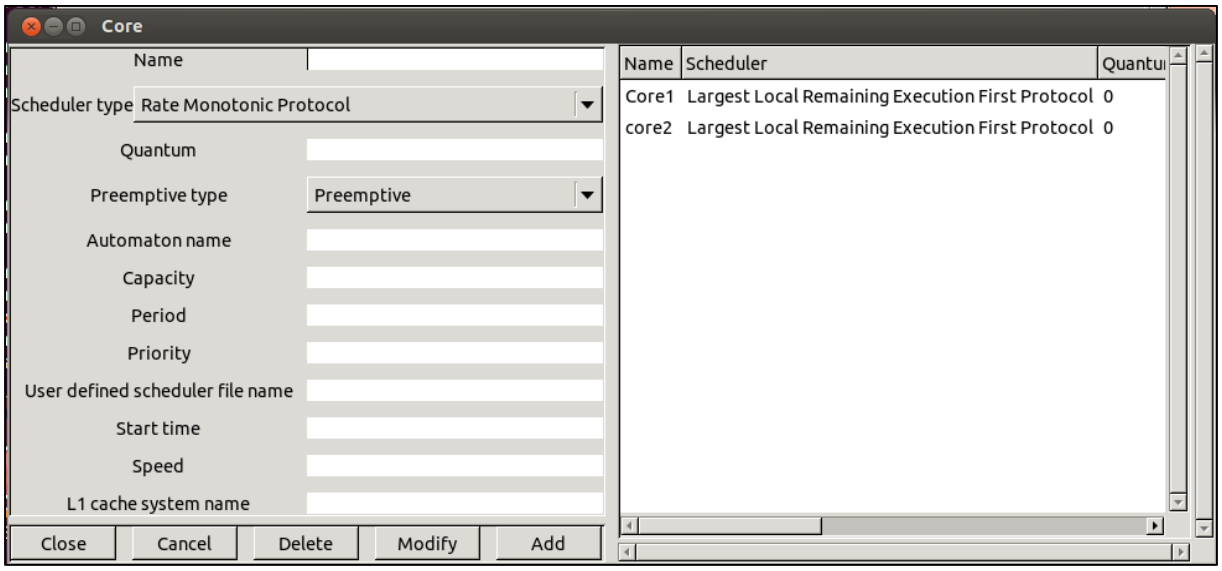

Figure 9: Interface d'ajout des cores dans Cheddar

Apres avoir ajouté les deux cores on doit maintenant déclarer un processeur avec deux cores comme indiqué dans la figure 10.

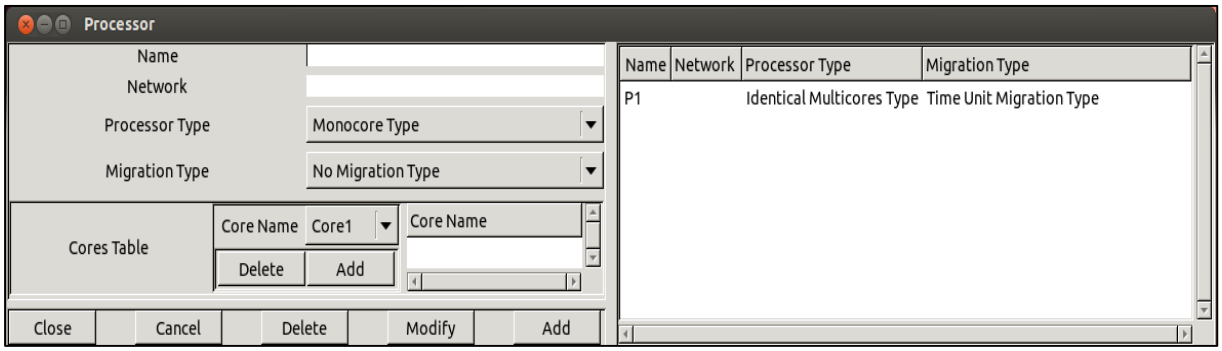

Figure 10: Déclaration des processeurs dans Cheddar

Finalement, on déclare les tâches de notre jeu de tâches comme c'est indiqué dans la figure 11.

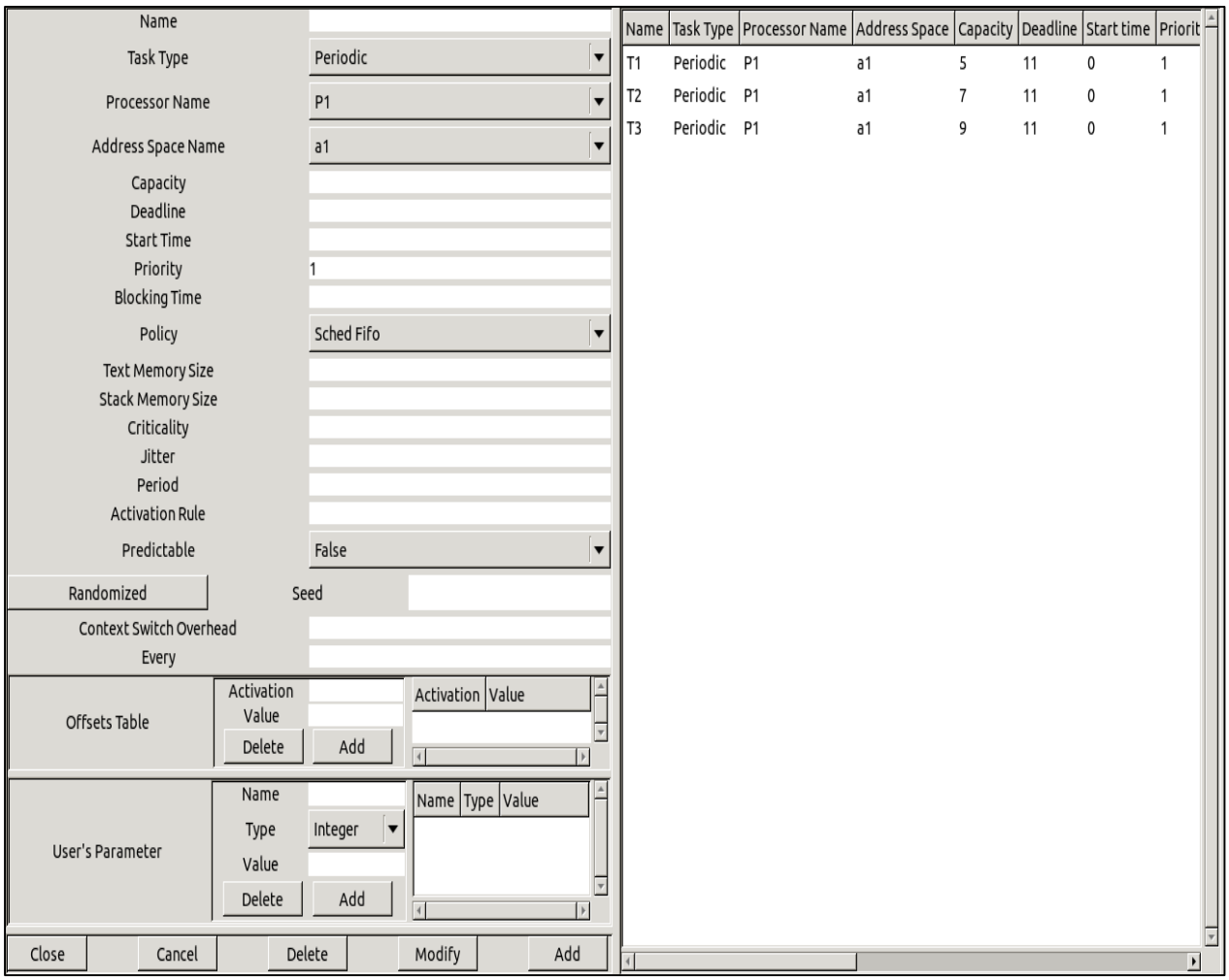

Figure 11: Ajout des tâches dans Cheddar

La figure 12 présente l'ordonnancement du premier jeu de tâches avec Cheddar selon l'algorithme LLREF.

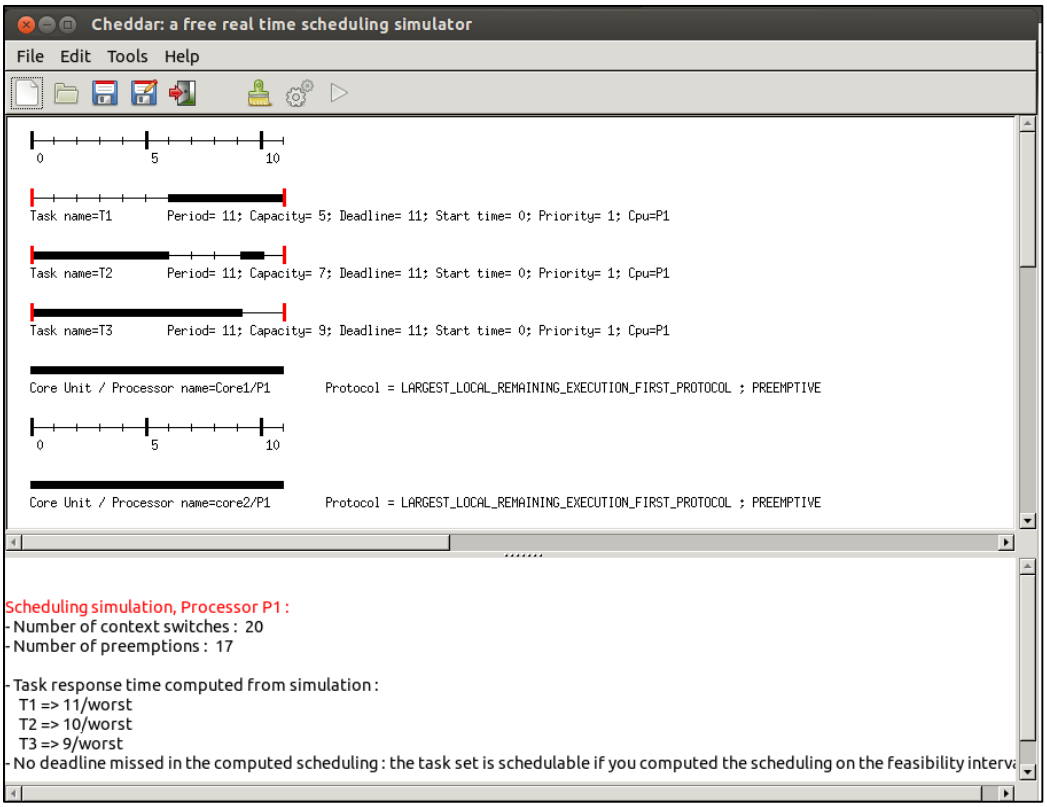

Figure 12: Ordonnancement du premier jeu de tâches selon LLREF avec Cheddar

La figure 13 présente l'ordonnancement du deuxième jeu de tâches avec Cheddar selon l'algorithme LLREF.

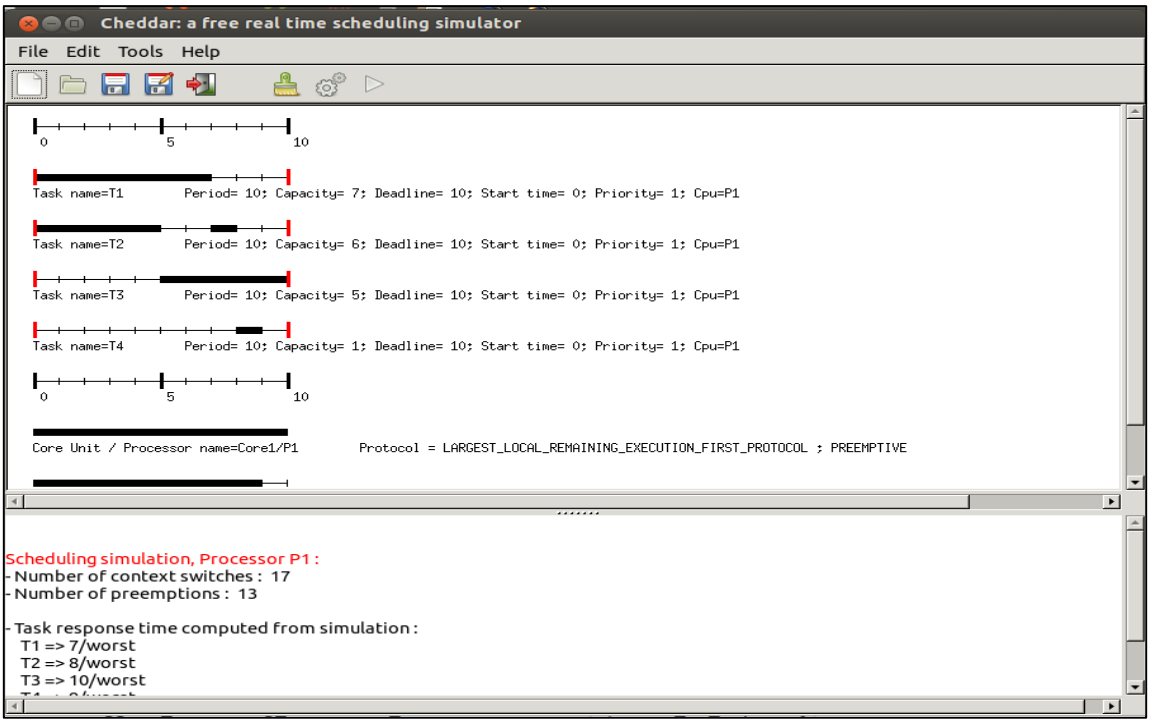

Figure 13: Ordonnancement du deuxième jeu de tâches selon LLREF avec Cheddar

Conclusion

Dans ce chapitre nous avons détaillé les technologies utilisées pour la réalisation de notre projet ainsi que les fonctionnalités de base de l'application à travers un ensemble de captures d'écran.

Conclusion générale

Le projet présenté dans ce rapport a pour but d'implémenter l'algorithme LLREF dans Cheddar.

En effet, le paramétrage et la configuration de cette solution améliore les possibilités de modification.

Comme perspectives d'amélioration de ce travail nous citons :

- Appliquer l'algorithme sur les tâches apériodique,
- Traiter le cas des contraintes de précédences,
- Traiter les tâches qui utilisent une / plusieurs ressources partagées.

Finalement, nous espérons que le travail accompli était à la hauteur des attentes de l'équipe qui a lancé le projet Cheddar. Ce projet est donc un avant-goût du travailleur d'un chercheur dans le domaine de l'informatique.

Bibliographie

[1] https://www.univ-brest.fr/menu/universite/Pr%C3%A9sentation+UBO/

[2] Cho, H., Ravindran, B., & Jensen, E. D. (2006, December). An optimal real-time scheduling algorithm for multiprocessors. In *Real-Time Systems Symposium, 2006. RTSS'06. 27th IEEE International* (pp. 101-110). IEEE.

[3] Singhoff, F., Legrand, J., Nana, L., & Marcé, L. (2004, November). Cheddar: a flexible real time scheduling framework. In *ACM SIGAda Ada Letters* (Vol. 24, No. 4, pp. 1-8). ACM.

[4] https://doc.ubuntu-fr.org/virtualbox

[5] https://fr.wikibooks.org/wiki/Programmation_Ada

[6] http://www.adacore.com/gnatpro/toolsuite/compilation/

[7] https://subversion.apache.org/

Annexe:

How to install Cheddar on Ubuntu

1- Install Virtual Box on Your Computer
2- Download the Cheddar Image here:
http://beru.univ-brest.fr/~singhoff/cheddar_ubuntu.vdi
3- Create a new Ubuntu 32 bits machine, and load the Cheddar image
4- You can now start your virtual machine. Password and login are both « cheddar »
5- Open a Terminal and go to the directory contained the Cheddar Sources (~/trunk/src/)
6- Execute the commande « svn update » to download the latest version of Cheddar
7- Modify the directories of CHEDDAR_DIR and GNAT2012 in example_bash.bash file (export CHEDDAR_DIR=/home/cheddar/trunk and export GNAT_DIR=/home/cheddar/GNAT2012). You should find the example_bash.bash file in the /home/cheddar/trunk folder.
7- Do source ../compile.bash to fix the environment
8- Do make to compile
9- Enjoy and don't forget to give us feedbacks

Known errors

1- When using svn, you may get the following message :
svn: symbol lookup error: svn: undefined symbol:
> svn_opt__eat_peg_revisions

To fix this, either update your os (by using sudo apt-get update)
or do not use svn in the terminal where you compile Cheddar
2- When using svn, you may get the following message :
“Password for '(null)' GNOME keyring:”

To fix this, use svn arguments :
--username ARG : specify a username ARG
--password ARG : specify a password ARG