6GEI264 − Projet de conception

Francis Emond
Malek Khattech
Mamadou Dia
Marc-André Jean

2017−04-12

Résumé

Document de spécification des besoins tel que présenté dans les gabarits de la norme ISO29110 pour le projet de conception hiver 2017.

Tables des matières

1 Introduction

Ce document décrit les systèmes à réaliser dans le cadre du projet de conception. Il présente le système éditeur, le système assembleur et le système micro-ordinateur (qui peuvent tous les deux contenir plusieurs autres sous-systèmes).

1.1 Acronymes

GUI : Graphical User Interface − Interface Utilisateur
PEP : Python Enhancement Proposal
PCONC : Projet de conception

1.2 Objectif

L'objectif de ce document est d'exposer le comportement du logiciel à réaliser dans le cadre du projet de conception du cours de vérification et validation des logiciels. Il présente avec minutie une liste des exigences fonctionnelles et non fonctionnelles contenues dans le document de vision rédigé préalablement en partenariat avec les différents intervenants du projet.

En d'autres termes, ce document décrit les fonctionnalités attendues de ce projet, soit :

- La conception d'un logiciel capable de compiler du code assembleur en code machine d'un micro-ordinateur.
- La conception d'un logiciel virtualisant un mini-ordinateur capable d'exécuter les programmes générés avec cet assembleur.

Chaque cas d'utilisation fait également l'objet d'une spécification, qui sera intégrée à ce document en annexe A.

1.3 Portée

Ce document décrit les différents composants nécessaires au fonctionnement de l'assembleur, du mini-ordinateur et de leurs fonctionnalités qui sont spécifiées dans le document de vision.

Un complément d'information peut également être retrouvé dans les notes de cours, ainsi que dans les documents des anciens laboratoires.

1.4 Références

PCONC-REF-01 : Site du cours 6GEI264 ; Vérification et validation logiciels
PCONC-REF-02 : Consignes du projet de conception
PCONC-REF-03 : PEP8 - Style Guide for Python code

1.5 Hypothèses et Dépendances

Pour réaliser l'assembleur, le logiciel du micro-ordinateur et l'éditeur, nous devons assumer certaines hypothèses.

Ces dernières peuvent se résumer comme suit:

H1 : Nous présumons qu'il n'y aura pas de changements au niveau des exigences qui pourraient avoir un impact considérable sur le développement de ce logiciel.

H2 : Nous présumons qu'il n'y aura pas des changements de matériel et de logiciel qui pourraient avoir un impact sur le développement de ce logiciel.

H3 : Tout changement approuvé dans le document de vision aura un impact sur ce document.

2 Survol du Modèle des Cas d'Utilisation

Le survol du modèle des cas d'utilisation illustre les principales fonctionnalités du système PCONC et est subdivisé en deux (2) sections:

- **Programmeur :** Cette section représente les différents cas d'utilisation (Écriture du code, compilation, sauvegarde, etc.) inhérents à un programmeur. (Figure I)
- **Utilisateur :** Cette section représente les différents cas d'utilisation (Chargement d'un programme, réinitialisation de l'exécution, etc.) inhérents à un utilisateur.. (Figure II)

2.1 Cas d'utilisation du programmeur

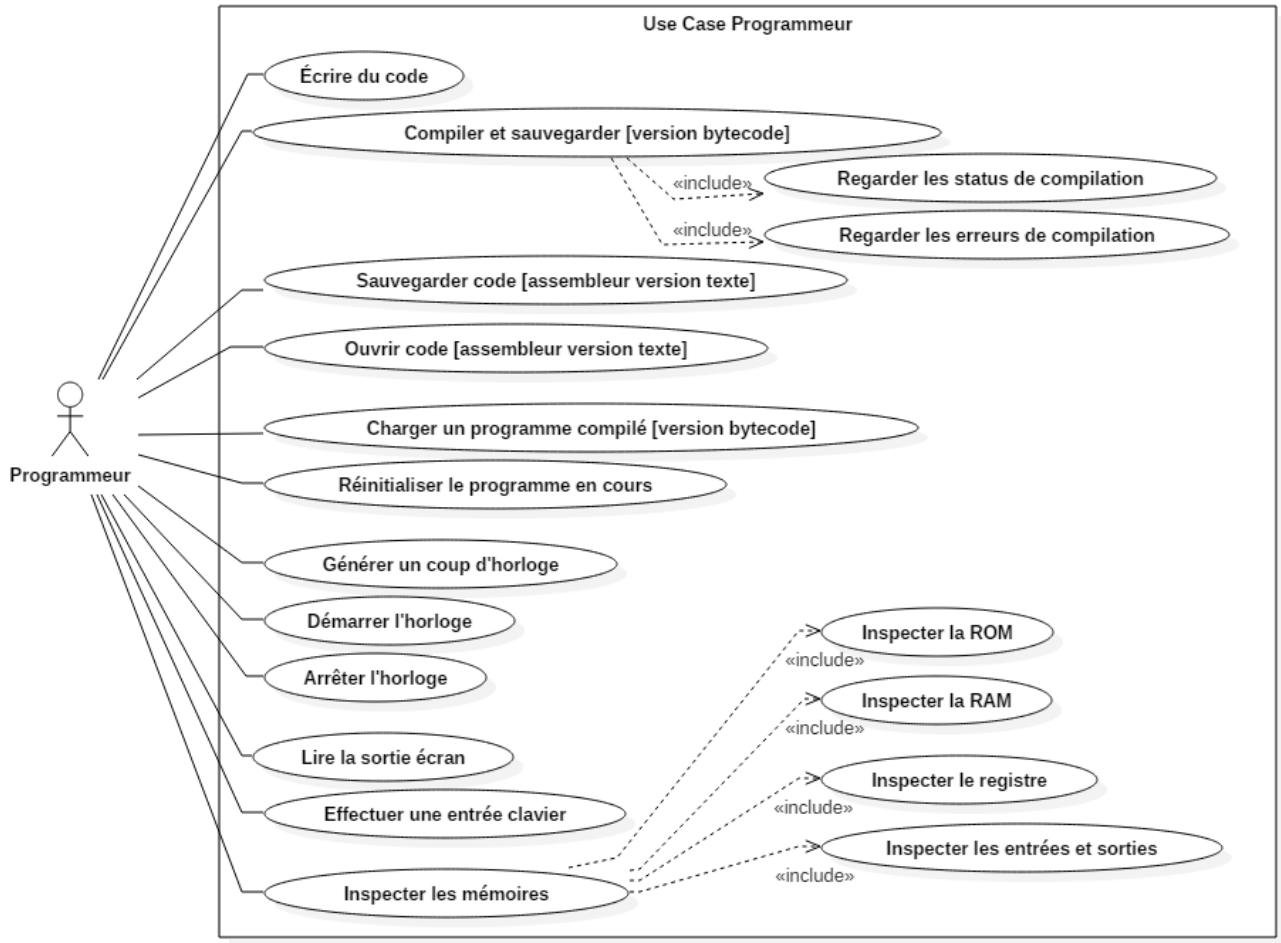

Figure I : Diagramme de cas d'utilisation du programmeur

2.2 Cas d'utilisation de l'utilisateur

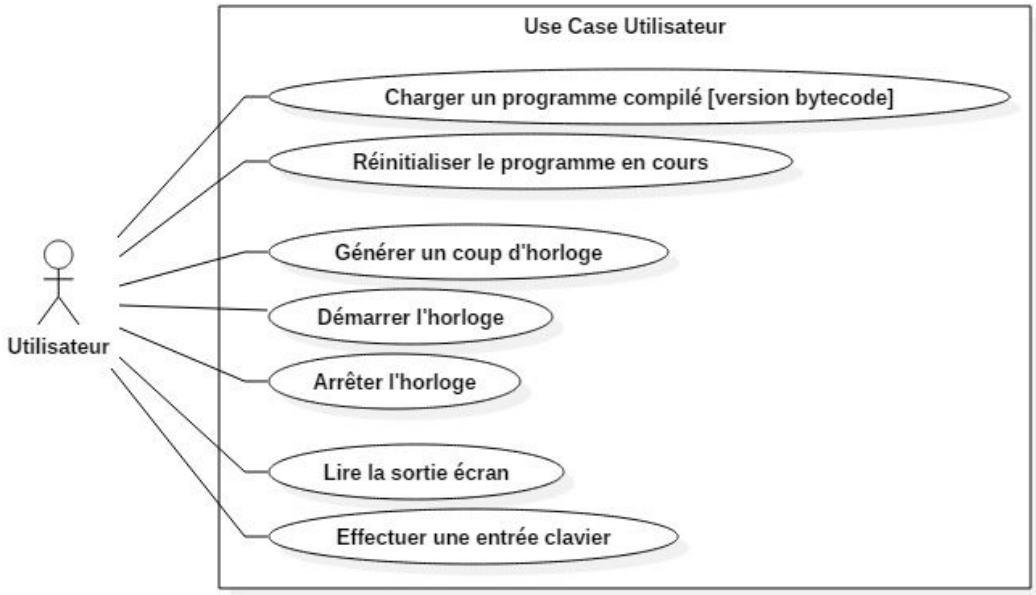

Figure II : Diagramme de cas d'utilisation de l'utilisateur

2.3 Description des cas d'utilisation

Chaque cas d'utilisation mentionné plus haut est décrit dans les lignes qui suivent:

PCONC-UCS-001 Écrire du code :

Le programmeur écrit ou modifie du code dans la zone d'édition de texte dans l'interface « Éditeur ».

Acteurs associés : Programmeur.

PCONC-UCS-002 Compiler et sauvegarder [version bytecode] :

Le programmeur appuie sur le bouton « Compiler et Sauvegarder » dans l'interface « Éditeur ». Cette action entraînera la compilation du code et de sa sauvegarde en version exécutable (bytecode). Une boite système sera ouverte pour que le programmeur puisse choisir le nom et l'endroit où sauvegarder le fichier.

Acteurs associés : Programmeur.

PCONC-UCS-003 Sauvegarder code [assembleur version texte] :

Le programmeur appuie sur le bouton « Sauvegarder code » dans l'interface « Éditeur ». Cette action entraînera la sauvegarde du code en version texte. Une boite système

sera ouverte pour que le programmeur puisse choisir le nom et l'endroit où sauvegarder le fichier.

Acteurs associés : Programmeur.

PCONC-UCS-004 Charger code [assembleur version texte] :

Le programmeur appuie sur le bouton « Ouvrir code » dans l'interface « Éditeur ». Cette action entraînera le chargement d'un code en version texte à partir d'un fichier existant. Une boite système sera ouverte pour que le programmeur puisse choisir le fichier à charger dans l'éditeur.

Acteurs associés : Programmeur.

PCONC-UCS-005 Charger un programme compilé [version bytecode] :

Le programmeur ou l'utilisateur appuie sur le bouton « Charger Programme » dans l'interface « Ordinateur ». Cette action entraînera le chargement d'un exécutable (bytecode) à partir d'un fichier existant. Une boite système sera ouverte pour que l'usager puisse choisir le fichier à charger dans l'éditeur.

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-006 Réinitialiser le programme en cours :

Le programmeur ou l'utilisateur appuie sur le bouton « Réinitialiser » dans l'interface « Ordinateur ». Cette action entraînera la réinitialisation de l'exécutable (bytecode) courant. Si aucun exécutable n'est chargé en mémoire dans l'ordinateur, cette action n'aura aucun effet.

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-007 Générer un coup d'horloge :

Le programmeur ou l'utilisateur appuie sur le bouton « Générer un coup d'horloge » dans l'interface « Ordinateur ». Cette action entraînera la génération d'un simple coup d'horloge dans l'ordinateur et avancera d'une instruction dans l'exécution de l'exécutable courant.

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-008 Démarrer l'horloge :

Le programmeur ou l'utilisateur appuie sur le bouton « Démarrer l'horloge » dans l'interface « Ordinateur ». Cette action activera l'horloge de l'ordinateur qui effectuerera des coups d'horloge d'une façon périodique à l'ordinateur.

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-009 Arrêter l'horloge :

Le programmeur ou l'utilisateur appuie sur le bouton « Arrêter l'horloge » dans l'interface « Ordinateur ». Cette action désactivera l'horloge de l'ordinateur.

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-010 Lire la sortie-écran :

Le programmeur ou l'utilisateur analyse et inspecte les sorties-écrans de son exécutable dans la zone de texte dans l'interface « Ordinateur ».

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-011 Effectuer une entrée clavier :

Le programmeur ou l'utilisateur place son curseur dans la zone d'édition pour les entrées clavier dans l'interface « Ordinateur ». Celui-ci effectue dans entrée clavier qui seront interceptés par l'exécutable.

Acteurs associés : Programmeur / Utilisateur.

PCONC-UCS-012 Inspecter les mémoires :

Le programmeur inspecte les mémoires à l'aide de la zone d'inspection des mémoires dans l'interface « Ordinateur ». Cette zone contient une liste déroulante qui affiche une plage d'adresse qui dépend de l'onglet choisi (RAM, ROM, etc.).

Acteurs associés : Programmeur.

3 Les acteurs

PCONC-ACT-001 Programmeur :

Le programmeur hérite de tous les usages du système de l'utilisateur. Il peut créer ou modifier du code assembleur dans l'éditeur. Ce code pourra, par la suite, être compilé pour générer un programme exécutable par le micro-ordinateur. Avec le système, le programmeur peut donc créer des programmes pour lui-même ou d'autres utilisateurs. Le programmeur peut exécuter et interagir avec des programmes avec le micro-ordinateur (comme l'utilisateur). Il peut aussi déboguer des programmes en utilisant la zone d'inspection des mémoires dans l'interface « Ordinateur » et en générant des coups d'horloge manuellement avec l'interface.

PCONC-ACT-002 Utilisateur :

L'utilisateur n'a théoriquement pas les connaissances en informatique et en assembleur d'un programmeur. Il se limite donc a exécuté et interagir avec des programmes déjà existants.

4 Les exigences

4.1 Les exigences fonctionnelles

4.1.1 Éditeur

PCONC-FSR-001 Le module d'édition doit contenir les fonctions suivantes :

- assembler() : compilation du programme de la source en code binaire
- load() : charge un fichier texte dans l'éditeur
- program() : copie le bytecode dans l'espace ROM de l'ordinateur
- executionMode() : lance l'exécution du programme
- clockMode() : une minuterie déclenche les changements d'étape
- buttonClick() : un clic sur un bouton déclenche les changements d'étape
- statusDisplay() : permet de visualiser l'état de l'ordinateur
- inputBuffer() : gère les entrées clavier destiné à l'ordinateur

4.1.2 Assembleur

PCONC-FSR-002 Le module assembleur doit contenir les fonctions suivantes :

- dataManipulation() : Move/Load
- arithmetic() : Add/Sub/Mul/Div
- logic() : And/Or/XOr/Not
- comparison() : Gt/Lt/Eq/GtE/LtE/Neq
- branch : JMP
- conditional : BRA non zero, BRZ zero, BRO overflow
- Other : NOP

PAS DE : sous-routines, utilisation de stack, interruption, etc.

PCONC-FSR-003 Modes d'adressages du module assembleur

- Immediate (data value written in code)
- Direct (memory address written in code)
- Indirect (memory address taken from a register value)

PAS DE : Relative/Index

4.1.3 Processeur

PCONC-FSR-004 Le module processeur doit contenir les fonctions suivantes :

- Fetch Instruction (recherche la prochaine instruction à exécuter)
- Decode Instruction (l'instruction est décodée)
- Read Adress (lit l'adresse de l'instruction et place sa valeur dans un registre)
- Execute

PCONC-FSR-005 Le processeur doit contenir les registres mémoire suivants :

- P : Program counter (16 bits)
- I : Instruction register (16 bits)

– A à D : General purpose register (4 x 16 bits)
– S : Status register (16 bits)

PCONC-FSR-006 Le Status register doit contenir les informations suivantes (de la gauche vers la droite) :

– Parity (le résultat de la dernière opération est pair ou impair)
– Sign (le résultat de la dernière opération est positif ou négatif)
– Carry (le résultat de la dernière opération a généré un dépassement)
– Zero (le résultat de la dernière opération est zéro)
– CND (le résultat de la dernière opération de comparaison est vrai ou faux

PCONC-FSR-007 Le bus de données doit contenir les éléments suivants :

– Data (16 bits)
– Adress (16 bits)
– Mode (wait, read, write) (16 bits)
– Clock

4.1.4 Fonctions assembleur à implémenter

Manipulation des données :

OP	REGISTER	VALUE	Description
DTA		16 bits value <value>	store <value> at current address
SET	A-D	16 bits value	set register <value>
LD	A-D	address	load content of memory at <address> into register A-D
ST	A-D	address	save content of register A-D to memory at <address>
MV	A-D	a-d	copy content of register a-d into register A-D

Arithmétique :

OP	REGISTER	REGISTER	Description
ADD	A-D	a-d	add content of register a-d to register A-D
SUB	A-D	a-d	subtract content of register a-d from register A-D
MUL	A-D	a-d	multiply content of register A-D by a-d
DIV	A-D	a-d	divide content of register A-D by a-d

Logique :

OP	REGISTER	REGISTER	Description
OR	A-D	a-d	A-D OU a-d résultat dans le registre A-D
AND	A-D	a-d	A-D ET a-d résultat dans le registre A-D
XOR	A-D	a-d	A-D OU EXCLUSIF a-d résultat dans le registre A-D
NOT	A-D	<none>	Négation du contenu du registre A-D

Comparaison :

OP	REGISTER	REGISTER	Description
LT	A-D	a-d	A-D est plus petit a-d
GT	A-D	a-d	A-D est plus grand a-d
LE	A-D	a-d	A-D est plus petit ou égal a-d
GE	A-D	a-d	A-D est plus grand ou égal a-d
EQ	A-D	a-d	A-D est égal a-d
EZ	A-D	<none>	A-D est égal à 0
NZ	A-D	<none>	A-D est différent de 0

Contrôle d'exécution :

OP	ADDRESS	REGISTER CONDITION	Description
JMP	address	<none>	branchement à l'adresse (aucune condition)
JMZ	address	S a le statut Zero à 1	branchement à l'adresse
JMO	address	S a le statut Overflow à 1	branchement à l'adresse
JMC	address	S a le statut CND à 1	branchement à l'adresse

Autres :

OP	ADDRESS	REGISTER CONDITION	Description
NOP	<none>	<none>	une opération qui n'effectue rien
HLT	<none>	<none>	arrête l'exécution du programme

4.2 Les exigences non fonctionnelles

4.2.1 Exigences minimales des modules

PCONC-NFCS-001 Éditeur :

Le module d'édition comprend minimalement une fenêtre d'édition avec les fonctions pour charger un programme à partir d'un fichier, lancer un programme avec sa sauvegarde ou sa transmission vers le ROM d'un ordinateur.

PCONC-NFCS-002 Assembleur :

Le module d'assemblage peut être réalisé de manière autonome et utilisable en ligne de commande.

PCONC-NFCS-003 Ordinateur :

Le module mini-ordinateur peut être lancé à partir de l'éditeur ou de manière autonome en ligne de commande. Son interface doit comprendre minimalement un écran et des fonctions pour charger un programme, pour le lancer, pour exécuter seulement une instruction de celui-ci et la saisie des touches du clavier.

PCONC-NFCS-004 Mémoire :

Toutes les adresses mémoires doivent contenir une valeur de 16 bits et doivent être réparties de la façon suivante :

- Adresses 0-32767 : Program (ROM) et les Entrées/Sorties
- Adresses 32768-65535 : RAM (Random Access Memory)

4.2.2 Particularités du code

PCONC-NFCS-004 Langage de programmation :

Le code du programme doit être rédigé dans le langage Python.

PCONC-NFCS-005 Tests :

Les fonctions doivent être accompagnées de tests nécessaires à la validation de celles-ci.

5 Documentation en direct pour l'utilisateur et exigence du système d'aide

PCONC-NFCS-006 :

Un manuel d'installation et de lancement du logiciel doit être fourni avec l'application.

PCONC-NFCS-007 :

Un manuel d'utilisation du logiciel doit être fourni avec l'application.

Aucun système d'aide en ligne n'est exigé.

Les exigences de documentation sont présentées dans les lignes qui suivent:

PCONC-DSR-001 Langage des documents :

Tous les documents doivent être rédigés en français.

PCONC-DSR-002 Format de fichier :

Tous les documents doivent être remis en format PDF, MSWord ou Texte (*.txt).

6 Contraintes de conception

Cette section présente les contraintes de conception.

PCONC-CC-001 : Le logiciel doit être codé en utilisant le langage Python.

PCONC-CC-002 : Le logiciel doit être compatible avec la version 2 ou 3 de Python.

PCONC-CC-003 : L'interface utilisateur doit utiliser la bibliothèque TKinter.

7 Composants achetés

Non applicable.

8 Interfaces

8.1 Interfaces Utilisateur

Les interfaces présentées ci-dessous ne sont que des suggestions. Elles ne représentent en aucun cas des modèles définitifs. Les interfaces devraient normalement être validées et devront avoir leurs propres ensembles de tests.

8.1.1 Interface Ordinateur

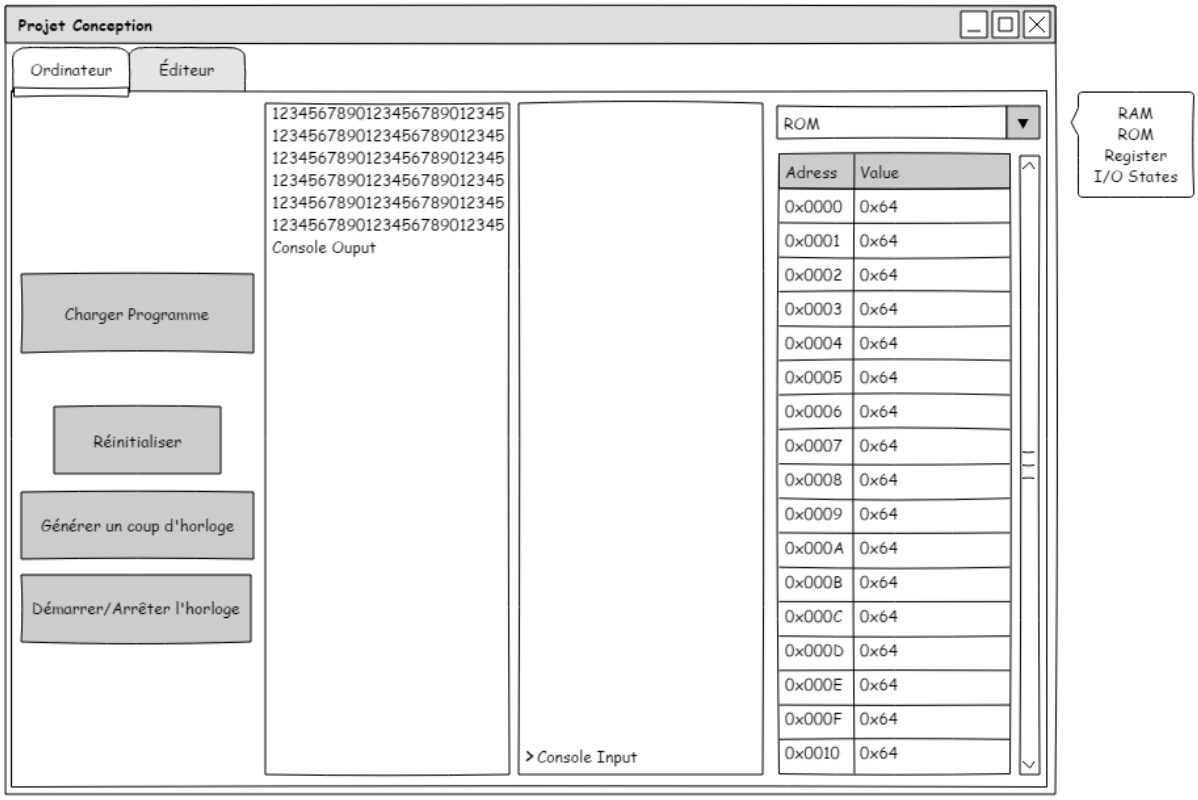

Figure III : Interface « Ordinateur »

8.1.2 Interface Éditeur

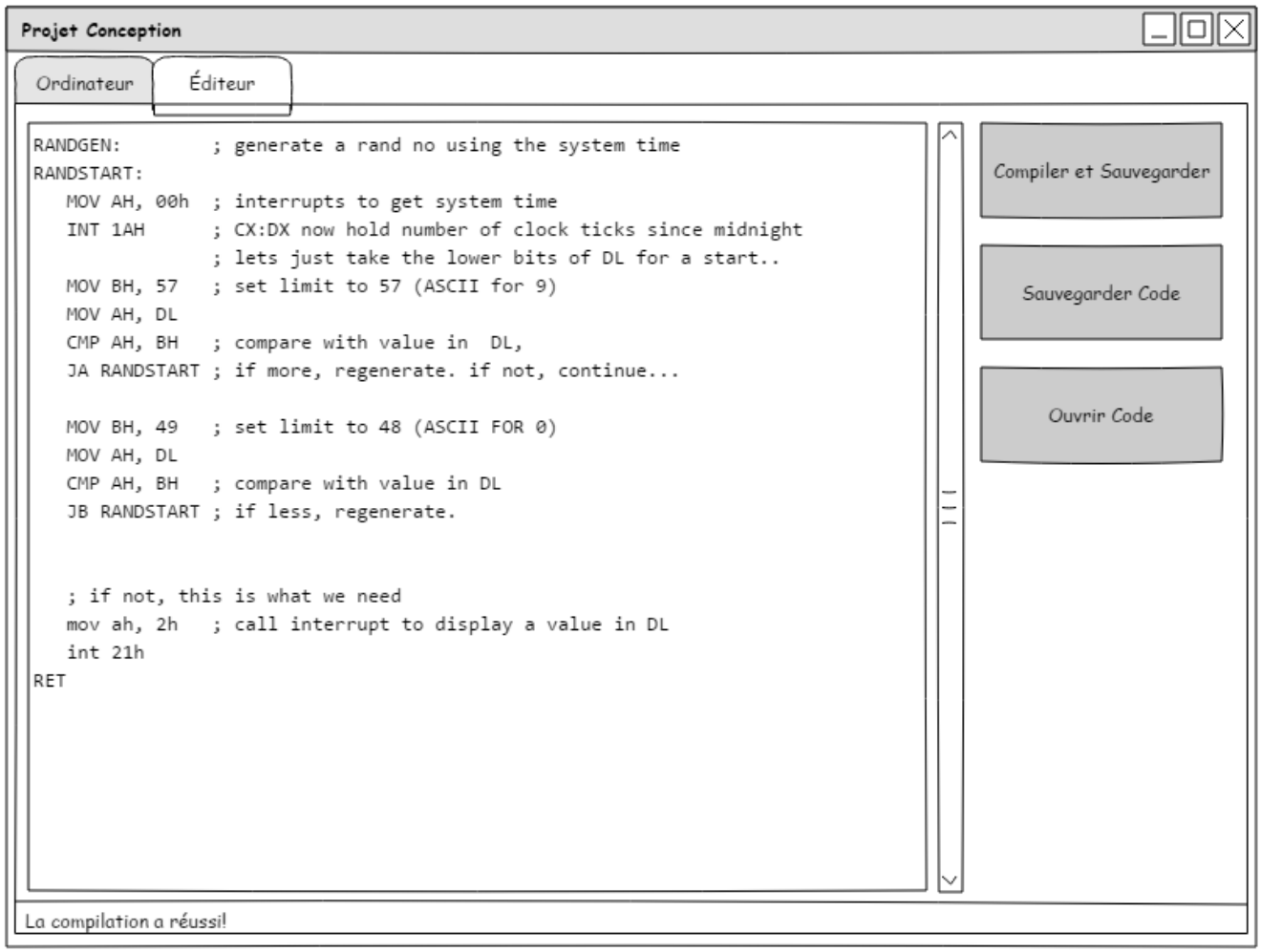

Figure IV : Fenêtre « Éditeur »

8.2 Interfaces Matérielles

L'application doit accepter des raccourcis clavier (spécifier dans le manuel d'utilisation du logiciel) correspondants aux touches de l'interface utilisateur directement au moyen du clavier.

L'application doit accepter un dispositif de pointage (souris, pavé tactile, etc.) qui activera les touches de l'interface utilisateur.

8.3 Interfaces Logicielles

Non applicable.

8.4 Interfaces de Communications

Non applicable.

9 Exigences de Licences

Non applicable.

10 Remarques légales, de droits d’auteur, et diverses

Non applicable.

11 Standards Applicables

- L’identification des documents doit suivre la nomenclature présentée en classe
- Les modèles de documents doivent être basés sur la norme ISO29110
- Le code doit suivre le standard PEP8 et PEP257 et ces standards doivent être identifiés dans la documentation.

A Développement des cas d'utilisation

A.1 PCONC-UCS-001 Écrire du code :

Brève Description

Le programmeur écrit ou modifie du code dans la zone d'édition de texte dans l'interface « Éditeur ».

Pré-condition

1. Être dans l'interface « Éditeur ».

Flux d'événements

Flux de Base

1. Le programmeur place le curseur à l'endroit souhaité dans la zone de texte d'édition du code
2. Le programmeur écrit ou modifie le code dans celle-ci en utilisant le langage et la syntaxe appropriée.

Flux alternatif

Non applicable.

Post conditions

1. Le système garde en mémoire le texte (code assembleur) de la zone d'édition en tout moment.

A.2 PCONC-UCS-002 Compiler et sauvegarder [version bytecode] :

Brève Description

Le programmeur appuie sur le bouton « Compiler et sauvegarder » dans l'interface « Éditeur ». Cette action entraînera la compilation du code et de sa sauvegarde en version exécutable (bytecode). Une boite système sera ouverte pour que le programmeur puisse choisir le nom et l'endroit où sauvegarder le fichier.

Pré-condition

1. Être dans l'interface « Éditeur ».

Flux d'événements

Flux de Base

1. Le programmeur appuie sur le bouton « Compiler et sauvegarder » dans l'interface « Éditeur »
2. Le système compile le fichier

3. Une boite système s'ouvre et le programmeur est invité à choisir le nom du fichier et l'endroit où le sauvegarder
4. Le programmeur confirme le nom et l'emplacement avec cette boite système
5. Le système enregistre le fichier exécutable
6. Le fichier est sauvegardé à l'emplacement et au nom choisi.

Flux alternatif

a. La compilation a échoué : Ce flux arrive à l'étape 2 du flux de base.

1. Une boite système « avertissement » notifie le programmeur que la compilation a échoué
2. L'interface « Éditeur » affiche sur la barre de statuts les informations de l'échec de la compilation
3. Le programmeur corrige son erreur de syntaxe ou autre
4. On retourne à l'étape 1 du flux de base.

b. Chemin invalide pour la sauvegarde : Ce flux arrive à l'étape 4 du flux de base.

1. Une boite système « avertissement » notifie le programmeur que le chemin invalide
2. Le programmeur doit rechoisir un emplacement correct avec la boite système
3. On retourne à l'étape 2 du flux de base.

c. Impossible d'ouvrir le fichier en écriture : Ce flux arrive à l'étape 5 du flux de base.

1. Une boite système « avertissement » notifie le programmeur que le fichier n'est pas accessible en écriture et lui demande de vérifier s'il est ouvert dans un autre programme
2. Le flux de base se termine.

Post conditions

1. Le fichier doit être enregistré à l'emplacement choisi et avec le nom choisit
2. Le fichier doit être intègre et avoir un format qui permet son utilisation.

A.3 PCONC-UCS-003 Sauvegarder [assembleur version texte] :

Brève Description

Le programmeur appuie sur le bouton « Sauvegarder code » dans l'interface « Éditeur ». Cette action entraînera la sauvegarde du code en version texte. Une boite système sera

ouverte pour que le programmeur puisse choisir le nom et l'endroit où sauvegarder le fichier.

Pré-condition

1. Être dans l'interface « Éditeur ».

Flux d'événements

Flux de Base

1. Le programmeur appuie sur le bouton « Sauvegarder code » dans l'interface « Éditeur »
2. Une boite système s'ouvre et le programmeur est invité à choisir le nom du fichier et l'endroit où le sauvegarder
3. Le programmeur confirme le nom et l'emplacement avec cette boite système
4. Le système enregistre le fichier texte de code assembleur
5. Le fichier est sauvegardé à l'emplacement et au nom choisi.

Flux alternatif

a. Chemin invalide pour la sauvegarde : Ce flux arrive à l'étape 4 du flux de base.

1. Une boite système « avertissement » notifie le programmeur que le chemin est invalide
2. Le programmeur doit rechoisir un emplacement correct avec la boite système
3. On retourne à l'étape 2 du flux de base.

b. Impossible d'ouvrir le fichier en écriture : Ce flux arrive à l'étape 5 du flux de base.

1. Une boite système « avertissement » notifie le programmeur que le fichier n'est pas accessible en écriture et lui demande de vérifier s'il est ouvert dans un autre programme
2. Le flux de base se termine.

Post conditions

1. Le fichier doit être enregistré à l'emplacement choisi et avec le nom choisit
2. Le fichier doit être intègre et avoir un format qui permet son utilisation.

A.4 PCONC-UCS-004 Charger [assembleur version texte] :

Brève Description

Le programmeur appuie sur le bouton « Charger code » dans l'interface « Éditeur ». Cette action entraînera le chargement d'un code en version texte à partir d'un fichier existant.

Une boite système sera ouverte pour que le programmeur puisse choisir le fichier à charger dans l'éditeur.

Pré-condition

1. Être dans l'interface « Éditeur ».

Flux d'événements

Flux de Base

1. Le programmeur appuie sur le bouton « Charger code » dans l'interface « Éditeur »
2. Une boite système s'ouvre et le programmeur est invité à choisir le fichier texte de code assembleur à charger
3. Le programmeur confirme le fichier à charger avec cette boite système
4. Le système charge le fichier et affiche son contenu dans la zone d'édition de texte (pour l'édition ou lecture du code)
5. Le programmeur a maintenant accès au code.

Flux alternatif

a. Chemin invalide pour le chargement : Ce flux arrive à l'étape 4 du flux de base.

1. Une boite système « avertissement » notifie le programmeur que le chemin est invalide
2. Le programmeur doit rechoisir un emplacement correct avec la boite système
3. On retourne à l'étape 2 du flux de base.

Post conditions

1. Le code doit être chargé dans la zone d'édition de code de l'interface « Éditeur ».

A.5 PCONC-UCS-005 Charger un programme compilé [version bytecode] :

Brève Description

Le programmeur ou l'utilisateur appuie sur le bouton « Charger Programme » dans l'interface « Ordinateur ». Cette action entraînera le chargement d'un exécutable (bytecode) à partir d'un fichier existant. Une boite système sera ouverte pour que l'usager puisse choisir le fichier à charger dans l'éditeur.

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Le fichier exécutable doit être bien formé et valide.

Flux d'événements

Flux de Base

1. Le programmeur appuie sur le bouton « Charger Programme » dans l'interface « Ordinateur »
2. Une boite système s'ouvre et l'usager est invité à choisir le fichier exécutable (bytecode)
3. Le programmeur confirme le fichier exécutable à charger avec cette boite système
4. Le système charge le fichier exécutable et prépare celui-ci dans le micro-ordinateur en mémoire
5. L'usager peut maintenant exécuter le programme chargé.

Flux alternatif

a. Chemin invalide pour le chargement : Ce flux arrive à l'étape 4 du flux de base.

1. Une boite système « avertissement » notifie à l'usager que le chemin est invalide
2. L'usager doit rechoisir un emplacement correct avec la boite système
3. On retourne à l'étape 2 du flux de base.

Post conditions

1. Le bytecode doit être chargé en mémoire dans l'espace mémoire approprié du micro-ordinateur.

A.6 PCONC-UCS-006 Réinitialiser le programme en cours :

Brève Description

Le programmeur ou l'utilisateur appuie sur le bouton « Réinitialiser » dans l'interface « Ordinateur ». Cette action entraînera la réinitialisation de l'exécutable (bytecode) courant. Si aucun exécutable n'est chargé en mémoire dans l'ordinateur, cette action n'aura aucun effet.

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur.

Flux d'événements

Flux de Base

1. L'usager appuie sur le bouton « Réinitialiser » dans l'interface « Ordinateur »
2. Le système restaure à l'état d'origine le programme ultérieurement chargé
3. L'usager peut maintenant exécuter le programme dans son état initial.

Flux alternatif

a. Aucun programme en mémoire dans le micro-ordinateur : Ce flux arrive à l'étape 2 du flux de base.

1. Une boite système « avertissement » notifie à l'usager qu'aucun programme n'est présentement chargé dans la mémoire du micro-ordinateur
2. Le flux de base se termine.

Post conditions

1. Le programme doit être dans son état par défaut dans la mémoire du micro-ordinateur.

A.7 PCONC-UCS-007 Générer un coup d'horloge :

Brève Description

Le programmeur ou l'utilisateur appuie sur le bouton « Générer un coup d'horloge » dans l'interface « Ordinateur ». Cette action entraînera la génération d'un simple coup d'horloge dans l'ordinateur et avancera d'une instruction dans l'exécution de l'exécutable courant.

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur.

Flux d'événements

Flux de Base

1. L'usager appuie sur le bouton « Générer un coup d'horloge » dans l'interface « Ordinateur »
2. Le système génère un simple coup d'horloge dans le micro-ordinateur ce qui fera avancer le programme courant d'une instruction
3. Le système exécute cette nouvelle instruction.

Flux alternatif

a. Aucun programme en mémoire dans le micro-ordinateur : Ce flux arrive à l'étape 2 du flux de base.

1. Une boite système « avertissement » notifie à l'usager qu'aucun programme n'est présentement chargé dans la mémoire du micro-ordinateur
2. Le flux de base se termine.

Post conditions

1. Le système doit avancer d'une instruction et exécuter celle-ci.

A.8 PCONC-UCS-008 Démarrer l'horloge :

Brève Description

Le programmeur ou l'utilisateur appuie sur le bouton « Démarrer l'horloge » dans l'interface « Ordinateur ». Cette action activera l'horloge de l'ordinateur qui effectuerera des coups d'horloge d'une façon périodique à l'ordinateur.

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur.

Flux d'événements

Flux de Base

1. L'usager appuie sur le bouton « Démarrer l'horloge » dans l'interface « Ordinateur »
2. Le système démarre le module horloge du micro-ordinateur qui effectuera des coups d'horloge de manière automatique et régulier
3. Le système exécute les instructions à chaque coup d'horloge.

Flux alternatif

a. Aucun programme en mémoire dans le micro-ordinateur : Ce flux arrive à l'étape 2 du flux de base.

1. Une boite système « avertissement » notifie à l'usager qu'aucun programme n'est présentement chargé dans la mémoire du micro-ordinateur
2. Le flux de base se termine.

Post conditions

1. Le module horloge du micro-ordinateur doit être activer
2. Le système doit exécuter l'instruction courante à chaque coup d'horloge.

A.9 PCONC-UCS-009 Arrêter l'horloge :

Brève Description

Le programmeur ou l'utilisateur appuie sur le bouton « Arrêter l'horloge » dans l'interface « Ordinateur ». Cette action désactivera l'horloge de l'ordinateur.

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur.

Flux d'événements

Flux de Base

1. L'usager appuie sur le bouton « Arrêter l'horloge » dans l'interface « Ordinateur »
2. Le système arrête le module horloge du micro-ordinateur qui n'effectuera plus de coups d'horloge de manière automatique et régulier.

Flux alternatif

a. **Aucun programme en mémoire dans le micro-ordinateur : Ce flux arrive à l'étape 2 du flux de base.**
    1. Une boite système « avertissement » notifie à l'usager qu'aucun programme n'est présentement chargé dans la mémoire du micro-ordinateur
    2. Le flux de base se termine.

Post conditions

1. Le module horloge du micro-ordinateur doit être désactiver.

A.10 PCONC-UCS-010 Lire la sortie-écran :

Brève Description

Le programmeur ou l'utilisateur analyse et inspecte les sorties-écrans de son exécutable dans la zone de texte dans l'interface « Ordinateur ».

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur
3. Le système doit avoir exécuté des instructions du programme en cours.

Flux d'événements

Flux de Base

1. Le système exécute une ou des instructions après un ou des coups d'horloge (généré manuellement ou automatiquement)
2. L'usager analyse et inspecte les sorties-écrans de l'exécutable dans la zone de texte (sortie-écran) de l'interface « ordinateur ».

Flux alternatif

Non applicable.

Post conditions

Non applicable.

A.11 PCONC-UCS-011 Effectuer une entrée clavier :

Brève Description

Le programmeur ou l'utilisateur place son curseur dans la zone d'édition pour les entrées clavier dans l'interface « Ordinateur ». Celui-ci effectue dans entrée clavier qui seront interceptés par l'exécutable.

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur.

Flux d'événements

Flux de Base

1. L'usager place son curseur dans la zone d'édition (pour les entrées clavier) de l'interface « Ordinateur »
2. L'usager écrit le texte qu'il souhaite envoyer au programme courant
3. L'usager valide son texte et l'envoie au système en appuyant sur la touche < Entrée >
4. Le système reçoit son texte et le place dans la plage mémoire appropriée du micro-ordinateur pour que le programme courant puisse l'utiliser
5. Le système efface le texte dans la zone d'édition.

Flux alternatif

a. Aucun programme en mémoire dans le micro-ordinateur : Ce flux arrive à l'étape 3 du flux de base.

1. Une boite système « avertissement » notifie à l'usager qu'aucun programme n'est présentement chargé dans la mémoire du micro-ordinateur
2. On retourne à l'étape 5 du flux de base.

Post conditions

1. Le texte entré par l'usager doit être dans la plage mémoire appropriée du micro-ordinateur pour que le programme puisse lire celle-ci.

A.12 PCONC-UCS-012 Inspecter les mémoires :

Brève Description

Le programmeur inspecte les mémoires à l'aide de la zone d'inspection des mémoires dans l'interface « Ordinateur ». Cette zone contient une liste déroulante qui affiche une plage d'adresse qui dépend de l'onglet choisi (RAM, ROM, etc.).

Pré-condition

1. Être dans l'interface « Ordinateur »
2. Un fichier exécutable doit être chargé dans le micro-ordinateur.

Flux d'événements

Flux de Base

1. L’usager choisit l’onglet approprié (dans la zone d’inspection des mémoires) dépendant de la plage qu’il souhaite jeter un coup d’oeil
2. L’usager utilise la barre de défilement (dans la zone d’inspection des mémoires) pour choisir la sous-plage pour laquelle il souhaite vérifier ou analyser.

Flux alternatif

Non applicable.

Post conditions

Non applicable.