Enseigner les bases de la commande logique avec HOME I/O et Scratch 3.0

Bernard Riera[1], David Annebicque[1], Idriss Moussa[2], Aristide Doucet[3], Fabien Emprin[4],
bernard.riera@univ-reims.fr
1 CReSTIC (EA 3804), UFR des Sciences Exactes et Naturelles, Université de Reims Champagne-Ardenne, Moulin de la Housse, BP 1039, 51687 Reims, France
2 IUT Reims, Université de Reims Champagne-Ardenne, Chemin des Rouliers 51687 Reims, France
3 Service des Usages du Numérique, Université de Reims Champagne-Ardenne, Moulin de la Housse, BP 1039, 51687 Reims, France
4 CEREP (EA 4692), Université de Reims Champagne-Ardenne, 23 rue Clément Ader 51100 Reims, France

RESUME : Le logiciel HOME I/O est le résultat d'un projet de R&D démarré en 2011 entre le laboratoire CReSTIC de l'Université de Reims Champagne-Ardenne (URCA) et Real Games, L'objectif était de concevoir une maison virtuelle pour l'enseignement des STEM (Sciences, Technologies, Ingénierie et Mathématiques) utilisable du collège à l'université. Lors des CETSIS en 2017 et 2021, HOME I/O et les premiers retours d'expérience ont été présentés. Cet article, décrit un module d'enseignement original d'initiation à la commande logique destiné à des étudiants en 2ème année de BUT « informatique » à l'IUT de Reims-Châlons-Charleville. HOME I/O, combiné avec Scratch 3 (logiciel utilisé au collège pour s'initier à la programmation), a été utilisé pour présenter les principaux concepts de la commande logique. L'approche pédagogique mise en œuvre, est basée sur un apprentissage par l'erreur, permettant aux étudiants d'appréhender, à partir des difficultés qu'ils ou elles rencontrent, les spécificités de l'informatique industrielle et de la commande logique. Les retours des étudiants obtenus au moyen de questionnaires anonymes sont globalement positifs.

Mots clés : commande logique, apprentissage par l'erreur, maison virtuelle, retour d'expérience.

1 INTRODUCTION

Le logiciel HOME I/O [5, 6] est le résultat d'un projet de R&D entre le CReSTIC de l'Université de Reims Champagne-Ardenne (URCA) et Real Games. L'objectif était de concevoir et d'amener une maison virtuelle dans la salle de classe, et utilisable du collège à l'université pour des enseignements en sciences et technologies en général, et en automatique (régulation, logique combinatoire et séquentielle) en particulier. Depuis 2017, HOME I/O peut être programmé avec Scratch [7, 8], le logiciel de programmation recommandé par l'Inspection Générale de l'Education Nationale pour l'initiation au codage au collège. En 2021, un serveur Scratch 3 a été développé et accessible par tous (https://crestic-scratch3.univ-reims.fr/). Celui-ci permet de développer en ligne des programmes Scratch 3 (contrairement à Scratch 2) pour piloter HOME I/O. Même si Scratch est considéré comme un moyen d'initier les jeunes à la programmation, nous présentons dans cet article comment Scratch 3, combiné avec HOME I/O, est utilisé dans un module d'initiation à la commande logique, afin d'introduire les principaux concepts des systèmes séquentiels et de l'informatique industrielle.

La première partie de l'article présente le module d'enseignement intitulé « Complément technologique ». Ce module de 18h TP a été suivi en 2022/2023, pour la première fois, par 54 étudiants en 2ème année de BUT "informatique » à l'IUT ("Institut de Technologie") de l'Université Reims Champagne Ardenne (URCA). L'objectif principal de ce module est de faire découvrir aux étudiants les bases de l'informatique industrielle pour être en mesure de développer un programme simple de commande logique, quel que soit le langage de programmation. Le principe de la pédagogie d'apprentissage par l'erreur a été utilisé. A travers les difficultés rencontrées, l'idée est de faire ressentir et comprendre aux étudiants les spécificités de l'informatique industrielle et de l'implémentation d'un contrôleur logique. Pour cela, les étudiants doivent, sans connaissance préalable, tenter d'implémenter des contrôleurs en Scratch 3, en vue de commander une lampe et la porte de garage de HOME I/O. Ils constatent alors par eux-mêmes que sans spécification formelle et sans méthode d'implémentation, le travail n'est pas aussi simple qu'il n'y paraît. La seconde partie de l'article traite de HOME I/O et du serveur Scratch 3 qui a été développé. La troisième partie de l'article présente les 2 problèmes de commande utilisés pour proposer un enseignement reposant sur un apprentissage à partir des erreurs. Une méthode d'implémentation originale d'une spécification GRAFCET (IEC 60848, 2002) [1] en langage Scratch 3 est également proposée et présentée pour la première fois. Enfin, dans la dernière partie de l'article, le retour d'expérience des étudiants montre que les retours des étudiants sont globalement positifs.

2 MODULE « COMPLEMENT TECHNOLOGIQUE » EN BUT 2 INFO

2.1 Objectif pédagogique

Le module de 18 heures TP « complément technologique » est destiné aux étudiants en 2ème année (semestre 4) du BUT « informatique » du parcours « Réalisation d'applications : conception, développement, validation ». Ce module entre dans le cadre de l'adaptation locale des programmes des BUT et a été mis en œuvre pour la première fois en 2022/2023. Toutefois, il fait suite à un module optionnel, suivi par une dizaine d'étudiants, d'initiation à la commande déve-

Article publié par EDP Sciences et disponible sur le site https://www.j3ea.org ou https://doi.org/10.1051/j3ea/20231025

loppé les années précédentes [7], dans lequel Scratch 2 était utilisé.
L'objectif principal est de donner aux étudiants, qui pour la majorité ne les possèdent pas, les bases de la commande et de l'informatique industrielle. En effet, un des défis majeurs de l'Industrie 4.0 est de rapprocher les mondes de l'IT (Information Technology) et de l'OT (Operational Technology). Le besoin de compétences multiples pour les techniciens et ingénieurs est de plus en plus important. Il convient donc de répondre aux enjeux à la fois techniques et numériques de la mise en place de la professionnalisation dans le cadre national de réforme globale de la formation (baccalauréat, Licence, Licences professionnelles, MEEF, etc.) afin de s'adapter aux exigences de l'usine du futur et, plus généralement, de notre société [2]. Ce module a été développé en ayant à l'esprit cet objectif et entre dans le cadre du projet PIA4 DemoES « DeMETeRE » dont l'URCA a été lauréate en 2021. L'idée est de permettre aux étudiants de prendre conscience des spécificités de l'informatique industrielle en général, et des contrôleurs logiques en particulier. Dans ce module, volontairement, nous ne nous concentrons pas sur les automates programmables industriels (API) et leur programmation avec les langages de la norme IEC 61131-3, mais sur deux exigences majeures lors de la conception et de la réalisation d'un contrôleur logique :

- La première étape de spécification formelle, avec une initiation au GRAFCET, absolument nécessaire avant de pouvoir coder le contrôleur.
- La deuxième étape d'implémentation pour être en mesure de coder un contrôleur à partir de la spécification formelle, quel que soit le langage de programmation et quel que soit le matériel (ARDUINO, Raspberry Pi, API...) ou le logiciel (C#, Scratch 3...).

Sur les 18 heures du module, les 12 dernières sont consacrées à la réalisation d'un programme et d'une IHM en C#, par les étudiants permettant de domotiser (éclairage, volets, porte de garage…) HOME I/O. Les 6 premières heures introductives, qui font l'objet de cet article, se déroulent dans le laboratoire d'automatismes de l'IUT de Reims pour permettre aux étudiants, en groupe TP, de découvrir « physiquement » des parties opératives et des parties commandes réelles.
Une approche pédagogique basée sur "l'apprentissage par l'erreur" a été utilisée pour permettre aux étudiants de saisir les spécificités de l'informatique industrielle et de la conception et de l'implémentation d'un contrôle logique,

2.2 Apprendre à partir des erreurs

L'apprentissage à partir des erreurs n'est pas nouveau. Pour John Dewey (1859-1952), « l'échec est instructif car la personne qui réfléchit vraiment apprend autant de ses échecs que de ses réussites ». L'échec est l'occasion pour les élèves de recevoir un retour d'information sur leurs points forts et leurs points à améliorer. Lorsqu'il est considéré comme un élément positif, constructif et essentiel de l'apprentissage, l'échec devient un outil pédagogique. De plus, faire des erreurs est un élément naturel du processus d'apprentissage. Beaucoup d'erreurs ne se produisent pas au hasard, mais proviennent d'idées fausses [3]. Néanmoins, bien que les erreurs doivent être éradiquées, différentes approches théoriques de l'apprentissage les considèrent comme bénéfiques pour les apprenants lorsqu'elles sont utilisées de manière constructive [9]. Cette approche pédagogique par l'erreur présente plusieurs avantages :
- L'apprentissage à partir des erreurs [4] favorise le développement personnel des apprenants car il donne aux élèves la possibilité de prendre des risques qu'ils n'auraient pas pris dans un contexte normal.
- L'apprentissage à partir des erreurs favorise la résolution de problèmes et la pensée critique. Avec la pédagogie par l'erreur, plutôt que de se contenter de suivre les instructions (tutoriels par exemple), ils doivent utiliser leurs compétences en matière de résolution de problèmes et de réflexion pour trouver par eux-mêmes une solution pratique et efficace.
- L'apprentissage à partir des erreurs renforce la rétention et la compréhension des connaissances.
- L'apprentissage à partir des erreurs supprime les limites créées par la peur de l'échec. Dans un environnement d'apprentissage sain où l'erreur n'est pas une tare, les étudiants n'ont pas peur de l'échec ou des conséquences de leurs erreurs. Les élèves seront davantage motivés pour faire des efforts et répondre aux questions, même s'ils ne sont pas certains de la bonne réponse.
- L'apprentissage à partir des erreurs permet aux apprenants de prendre conscience des problèmes rencontrés, et ainsi d'être davantage impliqués dans l'acquisition des connaissances et savoir-faire permettant de les résoudre.

Toutefois, en Automatique (au sens large), l'utilisation de systèmes réels, compte tenu des risques de casse, ne permet pas toujours facilement la mise en place d'une pédagogie d'apprentissage par l'erreur. En revanche, la simulation permet de proposer un environnement d'apprentissage sain où l'erreur est possible et ne constitue pas un danger. HOME I/O en est un exemple.

3 HOME I/O ET SCRATCH 3

3.1 La maison virtuelle HOME I/O

HOME I/O est un logiciel de simulation FPS (First Person Shooter) en temps réel (figure 1) d'une maison intelligente et de son environnement. Ce logiciel a été conçu pour couvrir un large éventail d'applications pédagogiques dans le domaine de la technologie et des sciences de l'ingénieur [6]. HOME I/O est un environnement d'apprentissage, d'expérimentation et de développement de projets dédié aux étudiants des collèges, des lycées et des universités. Tous les objets contrôlables peuvent être utilisés selon trois modes : câblé, console ou externe. En mode externe, les entrées et sorties de chaque objet (lumières, radiateurs, porte de garage, volets…) peuvent être utilisées par le biais d'un SDK qui ouvre le champ des applications de HOME

I/O. Cela permet simplement, par exemple, de contrôler la température d'une pièce avec LabView MATLAB, ou un API (Riera, 2017).

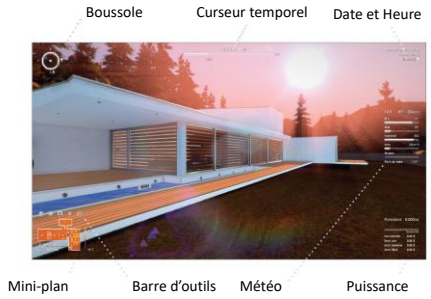

fig 1 : HOME I/O

HOME I/O dispose également d'un serveur web intégré permettant de récupérer les états des capteurs et d'agir sur les actionneurs à travers des requêtes http. Il est ainsi possible de développer, par exemple, des applications de type IoT. la méthode GET permet d'interroger le serveur web. Pour récupérer l'état de tous les capteurs, il faut utiliser la requête /poll. Chaque actionneur de HOME I/O peut être commandé au moyen d'une requête spécifique. La documentation est disponible à l'adresse suivante : https://docs.realgames.co/homeio/fr/web-server/.

3.2 Scratch 3

Scratch 3.0 est un environnement de programmation visuel conçu pour aider les enfants (et les adultes !) à apprendre à coder en s'amusant. C'est le logiciel qui a été retenu par le Ministère de l'Education Nationale en France pour initier les collégiens à la programmation. Scratch 3.0 a été développé par le groupe Lifelong Kindergarten du MIT, est gratuit et accessible en ligne (https://scratch.mit.edu/). L'interface utilisateur de Scratch 3.0 est basée sur des blocs de programmation qui peuvent être glissés et déposés pour créer des programmes. Les blocs sont organisés en catégories, telles que les mouvements, les événements, les sons et les costumes. Les programmes Scratch sont appelés « projets » et contiennent un ensemble de scripts. Ils peuvent être partagés en ligne. Scratch 3 a introduit de nombreuses améliorations par rapport à Scratch 2, notamment une interface utilisateur améliorée, de nouveaux blocs de programmation, une prise en charge de différents appareils et des améliorations de l'éditeur de code. La prise en charge des extensions et la compatibilité avec des périphériques externes ont également été améliorées dans Scratch 3.0. Toutes ces améliorations nous ont conduits en 2021 à développer un serveur web Scratch 3, ouvert à tous (écoles, grand public…), et hébergé à l'URCA (https://crestic-scratch3.univ-reims.fr/), intégrant une extension HOME I/O et permettant ainsi un pilotage de la maison virtuelle au moyen de blocs Scratch envoyant des requêtes au serveur web de HOME I/O (figure 2). Les états des capteurs sont actualisés toutes les 200 ms (requête /poll).

4 LES 2 PROBLEMES DE COMMANDE

Deux problèmes de commande simples ont été proposés aux étudiants : la lumière et la porte de garage (figure 3) pour leur faire prendre conscience des caractéristiques et des difficultés de réalisation d'un contrôleur logique.

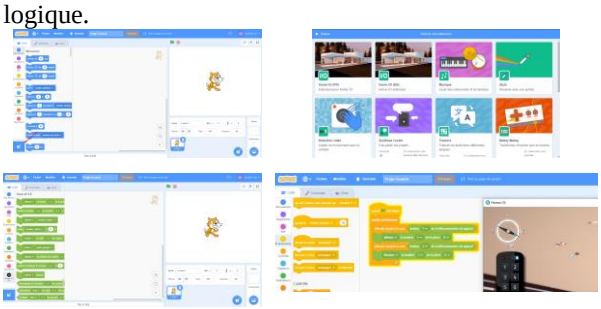

fig 2 : Scratch 3 et HOME I/O

fig 3 : La lumière et la porte de garage

4.1 Problème 1 : la lampe

Le cahier des charges est le suivant : un appui sur le bouton 1 de la télécommande doit allumer la lumière du garage si la lumière est éteinte, et l'éteindre si la lumière est allumée. Initialement, la lumière est éteinte. L'intérêt de ce premier exercice est de faire comprendre aux étudiants, par la difficulté qu'ils rencontrent à faire le programme en Scratch 3, les aspects spécifiques liés à la commande des systèmes séquentiels et de l'informatique industrielle :
- le concept d'entrées (E, capteurs) et de sorties (S, actionneurs) d'un contrôleur ;
- l'écart entre les spécifications textuelles et la programmation d'un contrôleur ;
- la notion de cycle ;
- la différence entre les événements (front montant ou front descendant) et les signaux (état).

La prise en main de HOME I/O et Scratch 3 ne pose pas de difficultés aux étudiants. En revanche, ils/elles rencontrent des problèmes dans la réalisation du programme avec les blocs Scratch. L'erreur la plus fréquente est la non prise en compte de la manière dont fonctionne le serveur web de HOME I/O (actions mémorisées) et l'envoi multiple de requêtes d'activation des actionneurs qui font tomber le serveur web, à la façon d'une cyber-attaque !

4.2 Problème 2 : la porte de garage

Le second cahier des charges concerne la commande de la porte de garage, et est plus complexe. Il est demandé de programmer le comportement suivant : un appui sur le bouton 1 de la télécommande ouvre la porte du ga-

rage, après un délai de 5 secondes en position ouverte, la porte se referme. Lors de la fermeture, si un appui sur le bouton 1 de la télécommande ou, si le capteur infrarouge de la porte de garage détecte un passage, la porte doit s'ouvrir à nouveau. L'intérêt majeur de ce second problème est de faire comprendre aux élèves à partir de leurs erreurs :

- la nécessité de méthodes de formalisation des spécifications, comme les Réseaux de Petri ou le GRAFCET (IEC 60848, 2002) ;
- le besoin de connaissance sur le système Partie Opérative/Partie Commande et de méthodologie pour être en mesure de programmer un contrôleur ;
- la difficulté de valider un contrôleur.

Après cette introduction, les étudiants, en raison des difficultés rencontrées et des erreurs commises, sont davantage intéressés par le contenu du cours. La deuxième partie de cette introduction de 6 heures traite du langage de spécification GRAFCET (IEC 60848, 2002). Les notions de base du GRAFCET sont uniquement présentées dans le cours : termes et définitions, représentation graphique des éléments (étapes, transitions, liens, réceptivités et actions continues et mémorisées), représentation graphique des structures séquentielles de base (cycle d'une séquence unique, sélection de séquences, et activation de séquences parallèles), et les principes généraux (règles de syntaxe et d'évolution). Nous insistons particulièrement sur la stabilité des étapes et l'importance d'éviter le parallélisme interprété. Toutefois, l'implémentation d'une spécification décrite par GRAFCET n'est pas incluse dans le champ d'application de la norme IEC 60848. La troisième partie du cours traite donc de l'implémentation.

La première solution présentée repose sur le fonctionnement cyclique et séquentiel d'un API, et est basée sur le calcul de la franchissabilité des transitions, des variables d'étape et des actions, en considérant les valeurs d'E/S constantes pendant un temps de cycle. Cette solution peut être utilisée quel que soit le matériel ou le logiciel du contrôleur et c'est celle que les étudiants implémenteront dans la deuxième partie du module lorsqu'ils développeront un contrôleur et une IHM en C#.

La seconde solution repose sur la connaissance des possibilités de Scratch 3. Aussi surprenant que cela puisse paraître, nous n'avons pas trouvé dans les ouvrages d'enseignement en technologie ou en sciences de l'ingénieur au collège ou au lycée de méthodologies rigoureuses de conception et d'implémentation d'un contrôleur avec Scratch. L'initiation au code avec Scratch repose uniquement sur la connaissance des organigrammes et des structures de contrôle (IF THEN ELSE, WHILE, FOR…). Cela est pour le moins surprenant car les extensions de Scratch pour piloter des robots LEGO par exemple sont très populaires au collège. Nous avons donc proposé une méthode d'implémentation d'un GRAFCET avec Scratch 3. Celle-ci fait l'objet du paragraphe suivant.

4.3 Implémentation d'un GRAFCET en Scratch 3

Scratch 3 intègre un système multitâche (ou « multi-threading ») permettant à plusieurs scripts (i.e. programmes) de s'exécuter simultanément sans interférer les uns avec les autres. Dans Scratch 3, chaque script s'exécute indépendamment des autres scripts dans le projet. Le multitâche avec Scratch 3 repose sur l'envoi et la réception de messages qui permettent de faire communiquer entre eux des scripts. Pour envoyer un message, on utilise le bloc "envoyer [nom du message]" dans les scripts. Le "nom du message" est un texte choisi pour identifier le message. Pour recevoir un message, on utilise le bloc "quand je reçois [nom du message]" dans les scripts. Les messages peuvent être utilisés par exemple pour transmettre des valeurs de variable entre les scripts.

Une méthode originale basée sur l'envoi de « messages », a été développée pour convertir un GRAFCET en scripts Scratch 3. L'idée est de découper le GRAFCET en séquences indépendantes (séquences linéaires, sélection de séquences et séquences simultanées) qui vont être représentées par un script et de les synchroniser au moyen de messages. Une sélection de séquences (divergence en OU) est réalisée en attendant qu'une des 2 réceptivités soit vraie. Le choix de la séquence s'effectue au moyen d'une structure SI-ALORS-SINON. La reprise de séquence (convergence en OU) est réalisée au moyen de l'envoi d'un message. Les séquences simultanées (divergence en ET) sont initiées au moyen d'un message envoyé simultanément à 2 scripts. Enfin, la synchronisation de séquences (convergence en ET) est réalisée au moyen de variables d'étapes. La figure 3 propose un GRAFCET « exemple » comprenant les différentes structures (ET, OU) et types d'actions (continue et mémorisée). Ce GRAFCET se décompose en 4 séquences. L'implémentation de celui-ci en scripts Scratch 3 est présentée figure 4. Le programme Scratch 3 est donc composé de 4 scripts (plus 1 pour l'initialisation) : séquence A (étapes 1 et 2), séquence B (étapes 3, 4 et 5), séquence C (étapes 7 et 8), et séquence D (étape 6). L'initialisation s'effectue au moyen d'un script permettant d'initialiser éventuellement des variables et d'activer l'étape initiale 1 du GRAFCET.

En appliquant cette méthode, il est simple de réaliser des programmes de commande en Scratch 3. On notera également que les programmes sont lisibles et représentent la structure du GRAFCET (au même titre s'un programme SFC). Toutefois, il est important de noter que le système multitâche de Scratch 3 peut être source de problèmes de performance si trop de scripts sont exécutés simultanément et qu'il faut gérer correctement l'envoi des requêtes. Les utilisateurs doivent donc en être conscients et optimiser leur code pour éviter tout ralentissement ou blocage.

La figure 6 indique une solution de GRAFCET et le programme Scratch 3 correspondant pour le problème de la lampe. On notera la façon dont les fronts montants sont gérés

La figure 7 propose une solution pour le problème de commande de la porte de garage. Le GRAFCET fait apparaitre une structure de type sélection de séquences qui est programmé comme indiqué précédemment.

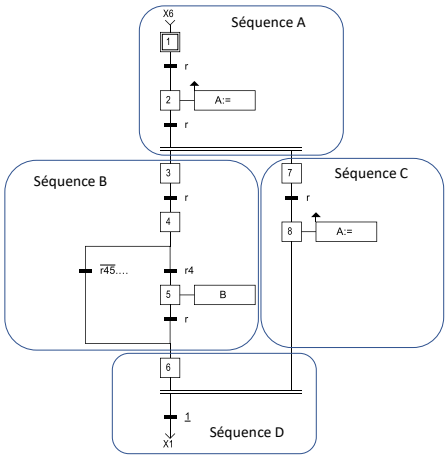

fig 4 : GRAFCET « exemple »

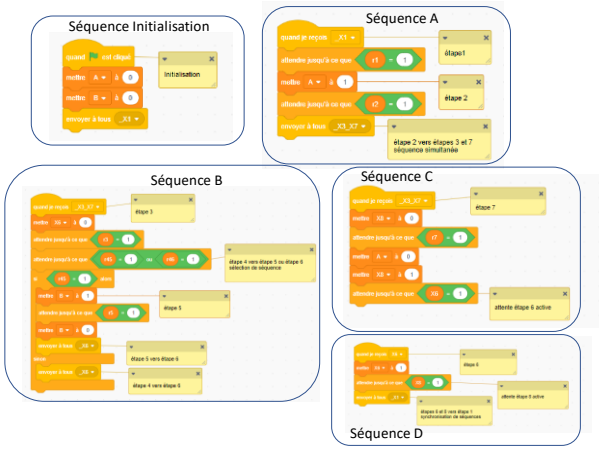

fig 5 : Implémentation en Scratch 3 du GRAFCET « exemple »

Après avoir présenté la méthodologie d'implémentation, les étudiants n'ont pas rencontré de difficulté à réaliser les programmes de commande de la lampe et de la porte de garage en Scratch 3.
Une évaluation du module a été réalisée auprès des étudiants. Une analyse des résultats obtenus est présentée dans la conclusion de l'article.

5 RETOUR D'EXPERIENCE

Un questionnaire a été envoyé aux étudiants pour qu'ils puissent évaluer de façon anonyme les 6 premières heures du module consacrées à la découverte de l'informatique industrielle et des contrôleurs logiques. L'objectif était de connaître entre autres leur sentiment vis-à-vis de la démarche pédagogique (apprendre par les erreurs, utilisation de HOME I/O et Scratch 3). Une réponse par groupe (monôme ou binôme) de TP était demandée. 29 réponses ont été obtenues pour les 7 questions posées :

Q1 : Un module d'initiation à « l'informatique industrielle » vous parait adapté à votre formation.
Q2 : J'ai compris ce qui été présenté.
Q3 : J'ai acquis de nouvelles compétences.
Q4 : J'ai apprécié la démarche pédagogique (expérimentations avec Scratch 3 + projet C#) au lieu d'un module d'enseignement habituel (CM, TD, TP).
Q5 : HOME I/O est un simulateur adapté à ce module d'enseignement.
Q6 : J'ai apprécié l'utilisation de HOME I/O.
Q7 : J'ai apprécié ce module d'enseignement.

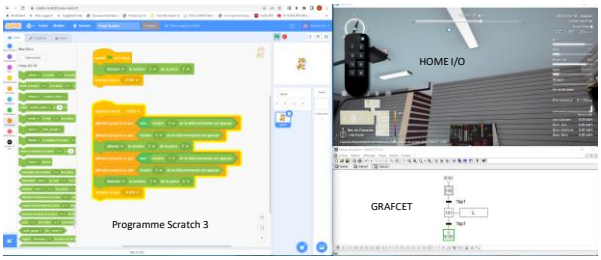

fig 6 : La commande de la lampe en Scratch 3

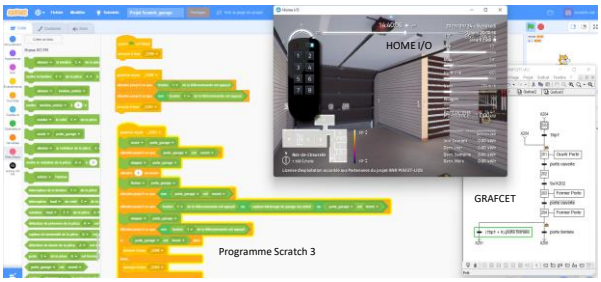

fig 7 : La commande du garage en Scratch 3

Le questionnaire est basé sur une échelle de Likert à 5 niveaux (de très défavorable à très favorable). Le niveau médian permet aux étudiants de ne pas se prononcer. La figure 8 présente les réponses cumulées pour chaque question. On notera que les avis positifs sont en vert, les avis négatifs en orange, et en blanc les étudiants qui ne se prononcent pas.
La figure 9 indique les réponses des étudiants par question. On constate que les étudiants utilisent peu le choix 3 : « ne se prononcent pas » sauf pour les questions 1 et 7 qui concernent davantage des avis sur la formation. On notera qu'il n'y a pas d'avis formellement négatif. Enfin, la figure 10 présente la boxplot des réponses aux questions. Les réponses aux questions 2, 4, 5, 6 et 7 montrent des avis favorables et uniquement très favorables pour les questions 5 et 6. Seule la question 1 a des réponses négatives (entre le premier et le 3ème quartile). Enfin, la question 3 a des réponses partagées. De façon synthétique, l'analyse des réponses au questionnaire semble indiquer que :
- HOME I/O est vu de façon très marquée comme un outil bien adapté et apprécié des étudiants.
- Les étudiants sont satisfaits de la démarche et du module.
- Ils ont en revanche du mal à juger de leurs acquisitions en termes de compétences et de l'intérêt d'une

initiation à l'informatique industrielle dans leur formation. Cela montre éventuellement un manque de recul sur la formation. Le message sur la convergence entre l'IT et l'OT n'est semble-t-il pas vraiment passé.

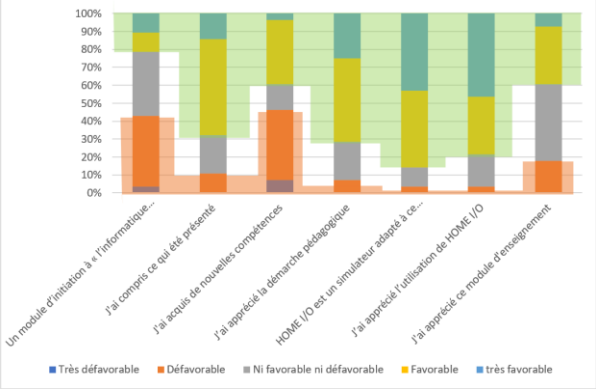

fig 8 : Réponses cumulées

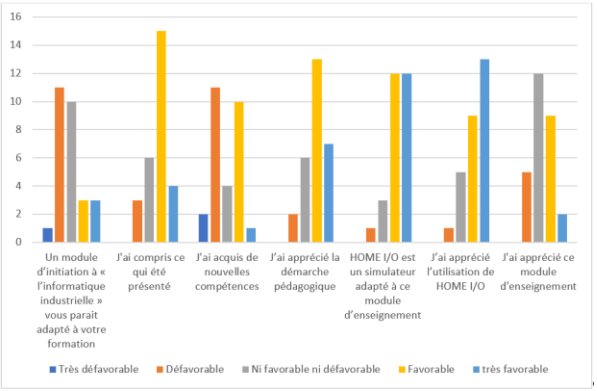

fig 9 : Réponses des étudiants par question

6 CONCLUSION

Dans cet article, un module d'initiation à la commande logique pour des étudiants en 2ème année de BUT « informatique », reposant sur l'utilisation de Scratch 3 et HOME I/O, a été présenté. Une pédagogie d'apprentissage par l'erreur a été mise en œuvre pour que les étudiants découvrent, et ressentent les spécificités de la commande logique. Les 2 problèmes de commande proposés sont simples à comprendre, tout comme le langage de programmation (Scratch 3), mais sans spécification formelle (GRAFCET) et sans une méthode de mise en œuvre, le travail n'est pas évident. Pour la première fois, une approche originale d'implémentation en Scratch 3 d'un GRAFCET a été proposée et détaillée dans l'article. Un des objectifs est de diffuser cette démarche dans l'enseignement secondaire. Les retours des étudiants au moyen de questionnaires montrent que l'utilisation de HOME I/O est adaptée à une pédagogie d'apprentissage par l'erreur. Ce module peut tout à fait être utilisé dans d'autres formations pour une initiation à la commande. Toutefois, les enseignants doivent, in fine, montrer aux étudiants que la conception d'un contrôleur basée uniquement sur la correction d'erreurs n'est pas la bonne façon de travailler, du tout !

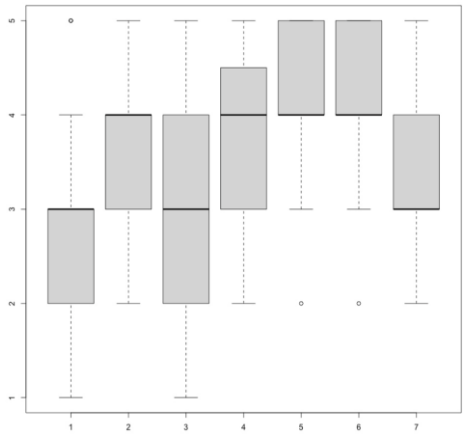

fig 10 : Boxplot des réponses aux questions

REMERCIEMENTS

Les travaux présentés dans cet article sont réalisés dans le cadre de DeMETeRE, projet lauréat du PIA4 DEMOES (Démonstrateur de l'enseignement supérieur) financé au titre du plan France 2030 - ANR-21-DMES-0011.

Bibliographie

[1] IEC INTERNATIONAL STANDARD 60848 (2002). Second edition 2002-02, GRAFCET specification language for sequential function charts. *Reference number CEI/IEC 60848: 2002*.

[2] Lamnabhi-Lagarrigue, F., Annaswamy, A., Engell, S., Isaksson, A., Khargonekar, P., Murray, R., Nijmeijer, H., Samad, T., Tilbury, D., Van den Hof, P. (2017) "Systems & Control for the future of humanity, research agenda: Current and future roles, impact and grand challenges". *Annual Reviews in Control 43 (2017) 1-64. [4]*.

[3] Nesher, P. (1987). Towards an Instructional Theory: The Role of Student`s Misconceptions. *For the Learning of Mathematics 7,3: pp 33-40, Publishing Association Montreal, Quebec, Canada.*

[4] Reuter, Y. (2020). La question de l'erreur–éléments pour un débat. *Recherches en didactiques, 2(9), 65-77*.

[5] Riera, B. Pichard, R., Philippot, A., Saddem, R., Gellot, F., Annebicque, D., Emprin, F. (2017). HOME I/O et FACTORY I/O : 2 logiciels innovants de simulation de PO pour la formation à l'automatique, *12ème Conférence CETSIS. Le Mans, France (mai 2017).*

[6] Riera, B., Vigario, B. (2017). "HOME I/O and FACTORY I/O: a virtual house and a virtual plant for control education", *IFAC-PapersOnLine, Volume 50, Issue 1, 2017, Pages 9144-9149, ISSN 2405-8963.*

[7] Riera, B. Philippot, A., Annebicque, D. (2019). « Teaching the first and only logic control course with HOME I/O and Scratch 2.0*", IFAC-PapersOnLine, Volume 52, Issue 9, 2019, Pages 109-114, ISSN 2405-8963.*

[8] Riera, B. Ranger, Saddem, R. Emprin, Chemla, J-P., Philippot, A. (2022). « Retour d'expérience et applications pédagogiques innovantes avec HOME I/O », *J3eA 21 2037 (2022), DOI: 10.1051/j3ea/20222037.*

[9] Wernecke, U., Schütte, K., Schwanewedel, J., Harms, U. (2018). "Enhancing Conceptual Knowledge of Energy in Biology with Incorrect Representations". *CBE Life Sci Educ. 2018 Spring;17(1). pii: ar5. doi: 10.1187/cbe.17-07-0133.*