



HAL
open science

Le dispositif Tabletop du projet AIDE : développement de son interface utilisateur.

Lola Denet

► To cite this version:

Lola Denet. Le dispositif Tabletop du projet AIDE : développement de son interface utilisateur.. [Rapport Technique] RT-0508, Inria Bordeaux Sud-Ouest. 2020, pp.14. hal-02935633

HAL Id: hal-02935633

<https://inria.hal.science/hal-02935633v1>

Submitted on 14 Sep 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License



Le dispositif Tabletop du projet AIDE : développement de son interface utilisateur.

Lola Denet

**TECHNICAL
REPORT**

N° 0508

Septembre 2020

Project-Team Mnemosyne



Le dispositif Tabletop du projet AIDE : développement de son interface utilisateur.

Lola Denet

Équipe-Projet Mnemosyne

Rapport technique n° 0508 — Septembre 2020 — 14 pages

Résumé : Le travail présenté ici consiste en la création d'une interface ludique et interactive pour un *serious game* qui est une des actions du projet : *Artificial Intelligence Devoted to Education* (AIDE). Ce projet a pour but de permettre le développement de la pensée computationnelle (ou pensée informatique) et d'en étudier les mécanismes d'apprentissage par des techniques, ou approches, basés sur les neurosciences cognitives et les sciences de l'éducation. L'expérience d'apprentissage passe par un jeu d'évasion (*escape game*) sur table utilisant des composants électroniques et des outils issus de la robotique. L'interface est visualisée sur un écran avec un processeur à bas coût. Elle se présente sous forme de pages Web. Le jeu communique avec l'interface en utilisant un serveur local et des techniques d'apprentissage automatique ou apprentissage machine (*machine learning*). Les interactions entre l'apprenant et la table de jeu sont filmées et analysées en temps réel, pour être affichée au sein de l'interface et enregistrées. Cette dernière est aussi chargée de transmettre les indications et les éléments scénarisés du jeu. La créativité, l'imagination et les compétences de l'apprenant sont ainsi sollicitées et évaluées.

Mots-clés : serious game, escape game, pensée computationnelle, apprentissage, compétences informatiques, neurosciences cognitives, science computationnelle de l'éducation.

**RESEARCH CENTRE
BORDEAUX – SUD-OUEST**

200 avenue de la Vieille Tour
33405 Talence Cedex

The AIDE project's Tabletop device: development of its user interface.

Abstract: The work presented here consists in the creation of a playful and interactive interface for a serious game part of the project: Artificial Intelligence Devoted to Education (AIDE). This project aims to help developing computational thinking from tangible and unplugged activity, while studying the human learning mechanisms considering models, based on cognitive neurosciences and educational science¹. The learning activity involves a tabletop escape game, collecting learning analytics using electronic components and software tools. The interface is visualized on a screen with a low cost processor. It is implemented as a set of web pages. The game communicates with the interface using a dedicated server, with machine learning algorithms. The interactions between the learner and the game table are filmed and analyzed in real time, to be displayed on the interface, and recorded. The latter is also responsible for transmitting the indications and scripted elements of the game. The creativity, imagination and skills of the learner are thus measured and evaluated.

Key-words: serious game, escape game, computational thinking, learning, computer skills, cognitive neuroscience, computational science of education.

1. <https://www.inria.fr/en/aide-new-exploratory-action-inria>

Table des matières

Introduction	1
1 Matériel et méthodes	2
1.1 Les pages web	2
1.1.1 Le scénario	2
1.1.2 Construction des pages	3
1.2 Implémentation des canevas	5
1.3 Architecture de l'interface	7
2 Résultats et discussion	9
Conclusion	13
Références	14

Introduction

Ce projet a pour but de permettre le développement de la pensée computationnelle (ou pensée informatique) et d'en étudier les mécanismes d'apprentissage par des techniques, ou approches, basés sur les neurosciences cognitives et les sciences de l'éducation ¹.

Le développement de la pensée computationnelle ^{2 3}, en tant que compétence du XXI^e siècle, est un élément capital pour vivre au quotidien et réussir dans une société où les technologies numériques font partie du quotidien. Il est important d'apprendre ou de ré-apprendre à identifier, comprendre et utiliser ces technologies afin de ne pas les subir mais aussi de remettre en question l'utilisation actuelle dans un but d'amélioration et de démocratisation. En effet, ces outils contribuent au développement des civilisations mais parfois au détriment de certaines valeurs humaines et aussi de la planète. L'impact environnemental et sociétal ne doivent pas être oubliés. La sensibilisation à ces sujets et l'initiation à la pensée computationnelle sont les enjeux de ce projet : revenir aux origines, et comprendre ce qui a conduit au développement et à l'utilisation de ces technologies afin d'en corriger les déviances. Un *teaser* reprenant ces éléments est disponible en ligne ⁴.

Pour enseigner, il est tout d'abord crucial de comprendre les mécanismes d'apprentissage. L'association des neurosciences cognitives et des sciences computationnelles de l'éducation permettent d'étudier cela. L'apprentissage par le jeu a déjà été exploré et a fait ses preuves.

Divya MENON et ses collaborateurs ont réalisé une importante revue de la littérature à ce sujet en 2019 [Menon *et al.*, 2019] également publiée dans une version française grand public [Romero *et al.*, 2018]. Dans celle-ci, il est démontré que les jeux d'évasion de table peuvent contribuer à maintenir l'implication du joueur dans les activités proposées. C'est-à-dire que cette méthode peut permettre aux apprenants de rester impliqués activement dans le processus d'apprentissage.

D'autre part, les composants de la compétence de pensée computationnelle sont explicités. Il se trouve que ces composants sont présents dans la plupart des jeux d'évasion ou de société. Il serait alors question d'utiliser ces outils pour transmettre, aux apprenants, les apprentissages nécessaires à l'acquisition de la pensée computationnelle. En effet, la pensée logique, la pensée algorithmique, la simulation, le débogage, la résolution de problèmes et la pensée collaborative sont des compétences développées et sollicitées dans ce type de jeux. Cependant, Divya MENON et ses collaborateurs [Menon *et al.*, 2019] ont identifié que les apprenants acquièrent ces composants de la pensée computationnelle sans pour autant qu'ils soient identifiés et conscientisés par ceux-ci. En effet, ils mobilisent des compétences capitales pour l'apprentissage de la programmation sans se rendre compte de ce qu'ils mobilisent. Or, il est important que les savoirs soient conscientisés pour être transférables. Ainsi, la métacognition, ou la mentalisation des processus mentaux, est un aspect à prendre en compte. Jean-Philippe LACHAUX explique en 2018 [Lachaux, 2018] que la métacognition est capitale pour l'apprentissage et la transférabilité des compétences. Il aborde également le fait que les compétences acquises par le jeu doivent faire l'objet d'une méthode et d'un vocabulaire adaptés afin de les transférer dans les apprentissages scolaires.

Divya MENON et ses collaborateurs [Menon *et al.*, 2019] ont mis en évidence que les activités physiques et cognitives peuvent être intégrées en utilisant la programmation. L'apprentissage de la programmation informatique et donc de la pensée computationnelle pourraient donc permettre d'y parvenir. Les apprenants seraient donc obligés d'utiliser leurs mains et leur esprit pour at-

1. <https://www.inria.fr/fr/aide-nouvelle-action-exploratoire-chez-inria>

2. <https://interstices.info/la-pensee-informatique/>

3. <https://project.inria.fr/classcode/mais-pourquoi-classcode-parle-de-pensee-informatique/>

4. <https://www.youtube.com/watch?v=K0JdRtHII1g&feature=youtu.be>

teindre les objectifs fixés. Cependant, la limite de ces techniques se trouve dans le moyen de les évaluer. En effet, dans les articles étudiés lors de cette revue, il est souvent question de difficultés à évaluer les activités des apprenants lors des jeux débranchés ainsi que leur impact. Ainsi, le projet AIDE a pour but de poursuivre ces recherches par le biais d'une action exploratoire.

Le projet AIDE⁵ s'inspire de cette revue pour créer un dispositif accessible chargé de mobiliser les différents composants de la pensée computationnelle. Les enjeux d'apprentissages sont clairs et il est évident que les méthodes d'évaluation des activités devront être explorées. Un autre enjeu de ce projet est l'accessibilité. Le dispositif utilisé est un jeu d'évasion de table conçu avec des matériaux à bas coût afin de permettre à n'importe quelle structure de concevoir elle-même son dispositif d'apprentissage de la pensée computationnelle. Ainsi, l'exclusion potentielle de certains groupes sociaux a été considérée lors de la conception afin que le milieu social ne soit pas un frein à l'apprentissage. Le rapport de recherche de Sabrina BARNABÉ et ses collaborateurs [Barnabé *et al.*, 2020], dont je suis co-autrice, développe avec précision ces éléments ainsi que le fonctionnement et les objectifs des différentes activités du dispositif.

En s'appuyant sur ces différentes notions, le dispositif utilise aussi bien des activités débranchées que des activités connectées. L'interface a un rôle capital dans les interactions avec l'apprenant. Elle contribue notamment à la métacognition, à l'accompagnement et à l'évaluation des activités. Cela sous-tend qu'elle doit respecter les concepts abordés précédemment en termes d'accessibilité, d'interactivité et de performance tout en restant agréable, simple et ludique.

La conception de cette interface sera développée dans la partie suivante de ce rapport. Ensuite, les résultats seront présentés et discutés, avant de conclure sur ce sujet.

1 Matériel et méthodes

Tous les fichiers sources et ressources multimédia associées de ce projet sont disponibles en ligne⁶ ainsi que la documentation générale⁷. Tous les fichiers composant l'interface sont également disponibles en ligne⁸ et sont documentés⁹. Ce rapport n'a donc pas vocation à reprendre la documentation mais développe le raisonnement et les méthodes de conception ainsi que les principes d'utilisation de l'interface.

1.1 Les pages web

1.1.1 Le scénario

Pour permettre l'immersion et l'implication de l'apprenant dans le jeu d'évasion, la création d'un scénario ludique paraissait essentielle. La collaboration entre les différents membres de l'équipe AIDE¹⁰ a permis de concevoir l'environnement, l'ambiance ainsi que les personnages chargés d'accompagner l'apprenant tout au long de son aventure. Ainsi, chaque protagoniste a une personnalité et un rôle bien identifiés.

D'autre part, le développement de l'interface prend en compte la taille du navigateur afin d'adapter son affichage. De plus, l'affichage des éléments sur les différentes pages a été pensé de sorte que l'ergonomie soit préservée puisque les interactions entre l'interface et l'apprenant se

5. <https://gitlab.inria.fr/line/aide-group/aide>

6. <https://gitlab.inria.fr/line/aide-group/aide/-/tree/master>

7. <http://aide-line.inria.fr/build/www/index.html>

8. <https://gitlab.inria.fr/line/aide-group/aide/-/tree/master/%40lola-denet/fr>

9. <http://aide-line.inria.fr/build/www-lola/index.html>

10. <https://team.inria.fr/mnemosyne/fr/aide/>

feront au travers d'un écran tactile (dimensions de l'écran en pixels : 1280x800).

Au début de la simulation, l'apprenant se trouve devant un château verrouillé dans lequel des génies de l'informatique ont été faits prisonniers par des monstres cruels qui refusent de partager les savoirs en lien avec la pensée computationnelle. Ces huit personnalités sont des grands noms de l'histoire de l'informatique.

Avant de pouvoir entrer dans le château, l'apprenant doit répondre à un questionnaire pour libérer Ozon, une dragonne qui va l'accompagner dans son périple. Ozon sera chargée de donner toutes les consignes de départ pour résoudre les énigmes du château. Elle sera accompagnée de quelques amis ayant des rôles bien distincts : Saturnin le lutin, Atlas le mage ancien et Stella l'elfe magique.

Deux savants (une femme et un homme) sont prisonniers dans chaque salle. La résolution d'une énigme permet de déverrouiller la porte menant à la salle suivante. Saturnin sera là pour apporter son aide et des indices à l'apprenant. A la fin de chaque épreuve, Atlas transmettra son savoir historique sur les savants libérés alors que Stella expliquera à l'apprenant les compétences qu'il a apprises et mobilisées au cours de l'épreuve.

Le scénario a été divisé en différentes parties selon les objectifs de chaque salle contenue dans le jeu d'évasion. Les éléments graphiques¹¹, les dialogues ainsi que les enregistrements audio¹² ont ensuite été créés en collaboration avec Mathieu MANRIQUE, moniteur éducateur diplômé d'Etat et artiste. Il a créé et dessiné les images de fond, les boutons ainsi que les personnages de l'interface et du *teaser*. De plus, il a prêté sa voix aux personnages masculins de l'interface.

Une vidéo tutorielle expliquant l'utilisation des boutons de l'interface et la présentation des personnages a été réalisée¹³.

1.1.2 Construction des pages

Le rapport de recherche de Sabrina BARNABÉ et ses collaborateurs [Barnabé *et al.*, 2020] détaillent la conception du jeu d'évasion ainsi que les objectifs de chaque salle. C'est sur cette base que les pages web de l'interface ont été élaborées.

Le jeu d'évasion comporte quatre salles auxquelles s'ajoutent une salle 0 permettant de recueillir des informations sur l'apprenant et une salle de fin servant de conclusion. Chaque activité est composée de deux fichiers `.html`. L'architecture des fichiers est détaillée dans le tableau de la figure 2 et dans le diagramme de la figure 4.

La salle 0 est chargée de l'introduction du scénario ainsi que de recueillir des informations sur l'apprenant par le biais d'un questionnaire. Dans le fichier `room0_1.html`, chaque question est sous forme d'`input` et a sa propre `div` de sorte qu'une seule question s'affiche à l'écran. Toutes les questions s'incluent dans un `form` de façon à recueillir les informations saisies dans les `input`. Au départ, toutes les divisions sont cachées. La première apparaît lorsque l'apprenant clique sur l'image cliquable de Ozon. Pour passer à la question suivante, la valeur saisie est contrôlée par la fonction `checkRequiredFields()` dans `interface.js` lorsque l'apprenant clique sur la flèche de droite. Cette fonction permet de vérifier au fur et à mesure des `input` qu'aucun champs n'est vide et que la valeur saisie est valide. Si c'est le cas, la division suivante s'affiche. Sinon, une alerte indique si la valeur est manquante ou incorrecte et met en surbrillance l'`input` concerné.

11. <https://gitlab.inria.fr/line/aide-group/aide/-/tree/master/%40lola-denet/images/mathieu>

12. <https://gitlab.inria.fr/line/aide-group/aide/-/tree/master/%40lola-denet/fr/audio>

13. <https://gitlab.inria.fr/line/aide-group/aide/-/tree/master/@lola-denet/fr/video>

Il est possible de revenir à la division précédente en cliquant sur la flèche de gauche. L'affichage des divisions est géré par les fonctions `showDiv()` et `showDivTechno()`. Elles sont appelées directement pour revenir en arrière mais sont appelées dans la fonction `checkRequiredFields()` pour avancer dans le questionnaire. La fonction `showDivTechno()` gère l'affichage d'une division en particulier car son affichage dépend de l'âge saisi par l'apprenant. Lorsque ce dernier est vérifié, la `divTechnoKids` ou la `divTechnoOlder` s'affiche. Une fois le dernier `input` rempli, un bouton s'affiche pour libérer Ozon, ouvrir le château et soumettre le formulaire. Les données sont ainsi sauveées dans un fichier `.json`.

Il existe différents moyens de remplir le questionnaire. En effet, lorsque la première division s'affiche, il faut saisir une adresse mail. Il est alors possible de cliquer sur le bouton "Charger". Cela provoque le remplissage automatique du formulaire si l'adresse mail existe déjà sur le serveur, c'est à dire, si l'apprenant a déjà répondu au questionnaire. Si c'est le cas et qu'aucune donnée n'est manquante, un nouveau bouton s'affiche pour permettre de passer le formulaire pour libérer Ozon directement. Si au moins une donnée est manquante ou invalide, l'autoremplissage fonctionnera mais ne permettra pas de passer le formulaire, il faudra alors repasser les divisions une à une mais une alerte affichera laquelle est incorrecte. L'autoremplissage du formulaire est également possible par l'url. En saisissant le nom des `input` et leurs valeurs dans la barre d'adresse, les champs se rempliront automatiquement. Tout cela est possible grâce aux fonctions `remanentForm()` et `remanentFormLoad()` dans `client.js`¹⁴. De même, c'est dans ce fichier que se trouve la fonction `fillString()` permettant d'anticiper le multilinguisme. En effet, le texte en français n'est pas saisi directement dans le fichier `.html`. Il est stocké sous forme de chaînes de caractères avec un identifiant unique dans `string.js`. La fonction est appelée dans une balise `<script>` afin d'afficher les chaînes de caractère au sein des balises où leurs identifiants sont saisis : `data-slug="id"`.

Une fois qu'Ozon est libérée et que le château est ouvert, un bouton s'affiche pour rediriger l'apprenant vers le fichier `room0_2.html` qui contient la vidéo tutorielle. Cette dernière a été entièrement montée lors de ce travail et explique l'utilisation des différents boutons de l'interface. Elle présente également tous les personnages de l'aventure.

Les salles 1 à 4 ont une structure similaire. Chacune possède deux fichiers `.html` et un fichier `.js` dédiés en plus des fichiers `.js` transversaux. Ceci est schématisé dans la figure 4. La partie 1 de chaque salle contient les éléments graphiques, les consignes et les canevas. Chacune a un fichier `.js` lié du même nom contenant les déclarations et initialisations des variables globales. Il est possible de les comparer aux fichiers `"header"` en `.h` que l'on trouve dans le langage de programmation C. La partie 2 de chaque salle contient uniquement les savoirs que l'apprenant doit acquérir (savoir historiques et compétences mobilisées). Les pages de ces parties 2 sont construites et les identifiants des chaînes de caractères créés dans `string.js` mais les contenus ne sont pas encore à jour car ils nécessitent d'être retravaillés en fonction des attentes de différentes activités. Il en est de même pour les contenus de des divisions d'aide et des indices dans les parties 1.

Dans ces salles, les consignes audio et dialogues sont lancés automatiquement à l'ouverture de la page web. Il est possible pour l'apprenant d'appuyer sur un bouton pour les afficher sous forme de texte à l'écran mais aussi de couper ou remettre le son en fonction de ses mécanismes d'apprentissage prévalents. En cliquant sur Ozon, il est possible de faire répéter la consigne autant de fois que nécessaire. Par la suite, il sera possible de recueillir les différents clics de l'apprenant pour l'étude statistique et la compréhension de ses besoins. Ainsi, il sera possible de savoir si un apprenant semble gêné par le son, combien de fois il fait répéter la consigne ou s'il a besoin de la lire. D'autre part, les divisions contenant les savoirs, les compétences, l'aide et les indices ne lancent pas d'audio de manière automatique. Il est nécessaire de cliquer sur le personnage

14. <https://gitlab.inria.fr/line/aide-group/aide/-/blob/master/src/client.js>

lié pour avoir la lecture à voix haute du contenu écrit. Cela permettra d'étudier si l'apprenant active le son délibérément en dehors des lectures audio automatiques.

La dernière salle contient la conclusion scénaristique de l'expérience et informe l'apprenant qu'il peut revenir sur le site web du projet pour accéder à un espace bonus. Cet espace est en cours de réalisation mais ne contient que des éléments scénarisés indépendants de l'étude menée. Cette partie sera intégralement réalisée par Mathieu MANRIQUE.

Enfin, un fichier `style.css`¹⁵ permet de gérer la mise en forme des différentes pages webs de l'interface. Ce fichier est indépendant de celui utilisé pour la mise en forme des autres pages web du projet.

1.2 Implémentation des canevas

Les salles 1 à 4 disposent chacune d'un canevas chargé de modéliser à l'écran les manipulations que l'apprenant fait sur le jeu de table représenté dans la figure 1. La caméra du dispositif filme les manipulations de l'apprenant et envoie au serveur ces informations afin de permettre l'affichage sur le canevas. Ceci utilise des techniques de *machine learning* et est en cours de développement. Des fonctions temporaires ont été créées pour simuler les interactions avec le plateau de jeu afin de pouvoir concevoir les canevas avant la fin du développement et de l'apprentissage machine.

Les images de fond des canevas sont des photos du plateau prises par la caméra puis améliorées par du traitement d'image. De même, les objets que l'apprenant positionne sur le canevas ont subi le même processus dans le but de permettre des analyses d'images ultérieures. Tout cela contribuera à l'apprentissage machine.

La salle 1 bénéficie d'un dispositif supplémentaire qui permet à l'apprenant de s'entraîner avant de lancer la simulation. En effet, un simulateur entièrement jouable en ligne et indépendant de la table de jeu a été implémenté. Dans ce simulateur, au lieu de capter la position du robot sur le plateau, l'apprenant utilise des boutons pour le déplacer directement sur le canevas. Des classes ont été utilisées pour créer le robot, les obstacles ou éléments de terrain. Ainsi, des notifications s'affichent si le robot percute un mur ou tombe dans un piège. De même, il devient invisible lorsqu'il passe sous le passage secret. Tout cela est possible grâce au fichier `simulator.js` qui contient les fonctions nécessaires à l'affichage et aux animations. Chaque bouton directionnel de la page web appelle les fonctions chargées de bouger et d'animer le robot. D'autre part, les alertes prévues initialement pour afficher les événements à l'écran ont été remplacées par des notifications de sorte que l'apprenant n'ait pas besoin de les fermer manuellement.

En ce qui concerne les simulations, la fonction `drawCanvas()` permet d'afficher et initialiser le canevas alors que la fonction `drawObjects()` affiche sur le canevas les objets transmis par la fonction `getTabletopObjects()`. L'algorithme veut que si un changement est détecté sur le plateau alors les nouvelles positions des objets sont transmises afin de modifier l'affichage sur le canevas. Cela est simulé par la fonction `hasTabletopChanged()` qui renvoie `true` en permanence pour les tests. `checkPosition()` renvoie des objets contenant des informations sur le statut du défi. Par exemple, les messages à afficher dans les notifications ainsi que l'état de réussite du défi. Si ce dernier a le statut `true`, alors la simulation prend fin automatiquement et un audio se lance pour signifier sa réussite à l'apprenant. Lors des tests, ces valeurs ont été modifiées. Actuellement, les notifications restent affichées en permanence et la simulation prend fin presque

15. <https://gitlab.inria.fr/line/aide-group/aide/-/blob/master/@lola-denet/style.css>

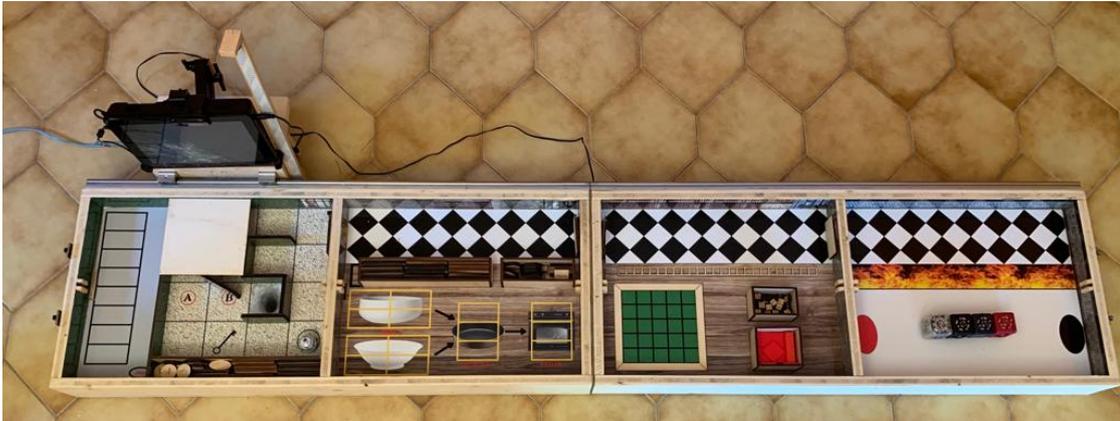


FIGURE 1 – Photographie du jeu d'évasion de table (*tabletop*) [Barnabé *et al.*, 2020]

Cette image est une photographie du jeu d'évasion sur table conçu par l'équipe du projet AIDE. Comme expliqué précédemment, il s'agit d'un dispositif à bas coût comprenant des composants électroniques et robotiques. Un petit écran équipé d'un processeur ainsi qu'une caméra se déplacent le long du plateau au fur et à mesure que l'apprenant avance dans les activités. Chaque boîte représente une salle distincte avec des objectifs, du matériel et des composants qui lui sont propres. La salle 1 est un labyrinthe avec un Ozobot représentant Ozon. L'apprenant doit créer une séquence d'instructions pour permettre au Ozobot d'arriver jusqu'à la sortie. La salle 2 est une cuisine dans laquelle l'apprenant devra assembler différents ingrédients (variables) pour réussir une recette de gâteau magique qui réveillera les génies endormis dans cette salle. La salle 3 est une bibliothèque avec une paroi magnétique invisible qui empêche les personnalités d'accéder à la sortie. L'apprenant devra alors transmettre et décoder plusieurs images ou séquences en langage machine afin de désactiver la paroi magnétique à l'aide du panneau de sécurité. La salle 4 est en feu et l'apprenant doit concevoir un véhicule autonome en assemblant les quatre CréaCubes devant lui afin de pouvoir s'échapper de la salle en flammes. L'assemblage avance automatiquement lorsque la configuration choisie est correcte.

aussitôt compte tenu des informations contenues dans les objets. Lorsque le développement de la table de jeu sera terminé, il sera nécessaire de retester tout cela afin de s'assurer que le comportement est conforme aux attentes. En effet, il est attendu que les notifications s'affichent de manière transitoire au moment où l'objet est transmis à l'interface. De même, l'interface ne recevra qu'une seule fois la valeur `true` pour la réussite du défi ce qui permettra de mieux apprécier les comportements intermédiaires de l'interface.

Les fonctions d'initialisation (`initPanel()`, `initGrid()`, `initWhiteBoard()` et `initControlPanel()`) permettent un positionnement optimisé des objets sur le canevas. En effet, une grille est créée de sorte que la *tabletop* transmette uniquement des coordonnées pour le positionnements des éléments. Ainsi, pour chaque couple de coordonnées sur les grilles, il existe des valeurs `x` et `y` prédéfinies en pixels et stockées dans des objets.

Les fonctions `initGame()` et `playAgain()` sont les fonctions principales des simulations. Elles permettent de démarrer les simulations, de gérer les modifications d'affichages et des événements par appels des autres fonctions. Les vérifications de l'état de la table de jeu sont faites à intervalles réguliers grâce à la méthode `setInterval()`. Cette méthode crée une boucle infinie qui ne peut être stoppée que lorsque l'interface reçoit la réussite du défi par la table de jeu (ou ici, lorsque `done` est `true`).

1.3 Architecture de l'interface

L'architecture de l'interface est présentée dans cette partie sous forme de deux tableaux et d'un diagramme. Les différents fichiers impliqués dans le fonctionnement de l'interface y sont expliqués.

La figure 2 liste les fichiers permettant l'affichage de l'interface.

Fichiers	Type du fichier	Nombre de lignes	Utilité
index.html	HTML	25	Donne accès : - au jeu, - aux raccourcis vers les salles, - à l'espace bonus.
room0_1.html	HTML	226	Contient : - le questionnaire de départ, - la mise en place du scénario. Permet : de recueillir des informations sur l'apprenant.
room0_2.html	HTML	41	Contient : la vidéo tutorielle.
room1_1.html room2_1.html room3_1.html room4_1.html	HTML	161 158 142 147	Permettent : - lancer les simulations, - d'interagir avec l'interface. Contiennent : - les consignes des défis (Ozon), - les éléments d'aide et indices (Saturnin).
room1_2.html room2_2.html room3_2.html room4_2.html	HTML	90 89 89 89	Contiennent : - les savoirs historiques (Atlas), - les compétences acquises (Stella).
roomEnd.html	HTML	59	Contient : la conclusion scénarisée de l'expérience.
style.css	CSS	397	Permet : la mise en forme des pages web.

FIGURE 2 – Architecture de l'interface : partie affichage.

Ce tableau contient la liste des fichiers impliqués dans l'affichage de l'interface web. Il regroupe, dans la première colonne, 12 fichiers .html et 1 fichier .css. La deuxième colonne précise le type du fichier et ainsi le langage de programmation utilisé. La troisième colonne indique le nombre de lignes de codes présentes dans chaque fichier. La dernière colonne résume le rôle de chaque fichier. Certains fichiers sont regroupés sur une même ligne du tableau car ils ont le même rôle mais aussi une structure semblable.

La figure 3 liste les fichiers permettant de rendre l'interface fonctionnelle.

Fichiers	Nombre de lignes	Nombre de fonctions et classes	Utilité
room1_1.js room2_1.js room3_1.js room4_1.js	39 27 27 27	-	Contiennent : les déclarations et initialisations des variables globales propres aux canevas de chaque salle. Sont liés : aux fichiers .html du même nom.
string.js	108	-	Contient : les chaînes de caractères affichées dans les pages de l'interface. Permet : d'anticiper le multilinguisme de l'interface. Est lié : à tous les fichiers .html.
interface.js	535	17 fonctions	Contient les fonctions nécessaires : - à l'adaptation de l'affichage du questionnaire et des différents éléments des salles, - à la vérification du formulaire, - à la gestion de l'audio (démarrage, arrêt, répétition), - à la simulation des interactions avec le jeu de table (fonctions provisoires), - à l'initialisation, l'affichage et l'actualisation du canevas et des objets qu'il contient. Est lié : à tous les fichiers .html.
simulator.js	329	13 fonctions 2 classes	Contient les classes et fonctions nécessaires : à l'initialisation, l'affichage et l'actualisation du canevas et des objets qu'il contient. Permet : de s'entraîner avant le défi de la salle 1 sur un simulateur totalement indépendant du jeu de table (entièrement jouable en ligne). Est lié : au fichier room1_1.html.

FIGURE 3 – Architecture de l'interface : partie fonctionnelle.

Ce tableau contient la liste des fichiers composant la partie "fonctionnelle" de l'interface. La première colonne regroupent 7 fichiers .js qui utilisent donc tous le langage de programmation JavaScript. La deuxième colonne indique le nombre de lignes de code présentes dans chaque fichier. La troisième colonne montre le nombre de fonctions et de classes pour chaque fichier. La dernière colonne résume le rôle de chaque fichier mais aussi à quels fichiers .html de l'interface ils sont liés. Certains fichiers sont regroupés sur une même ligne du tableau car ils ont le même rôle mais aussi une structure semblable.

La figure 4 montre les interactions entre les différents fichiers. C'est un schéma qui vient compléter l'architecture précédemment expliquée.

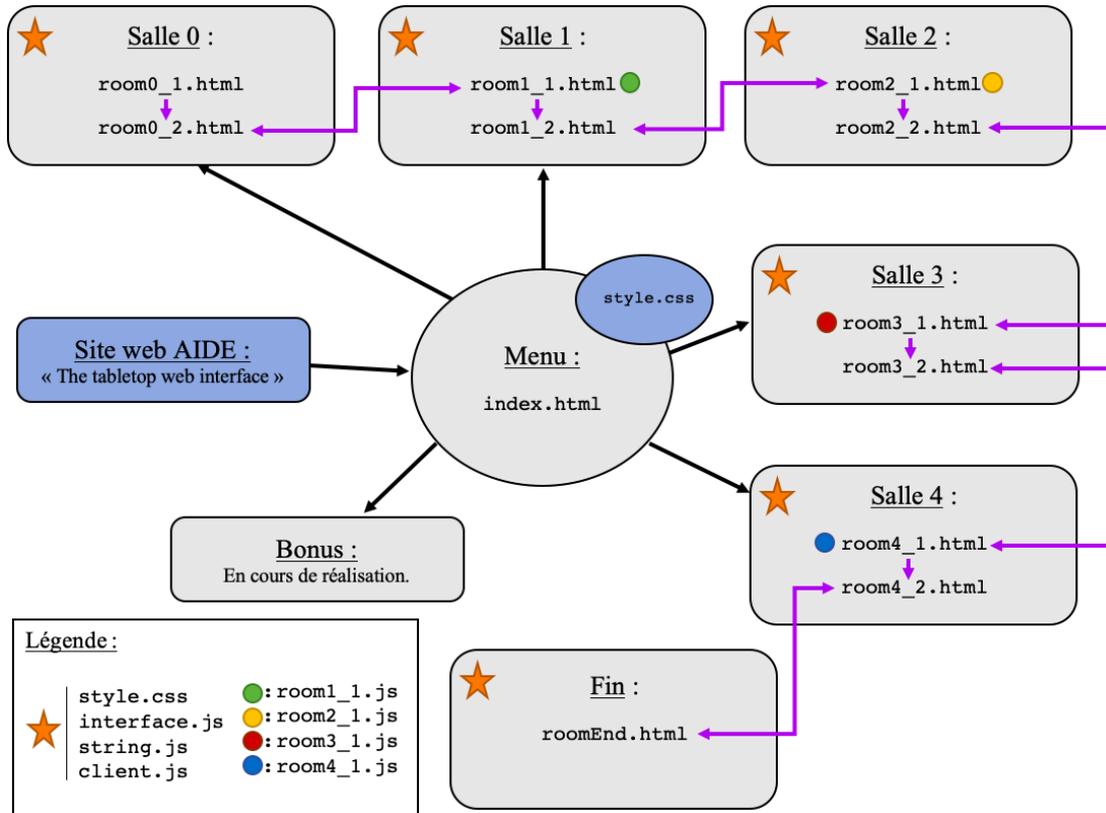


FIGURE 4 – Diagramme des interactions des fichiers de l'interface.

Ce diagramme représente tous les fichiers impliqués dans l'interface ainsi que leurs interactions. Les cases ayant un fond gris représentent les différentes salles affichées à l'écran par l'interface. Elles ont été entièrement conçues lors de ce travail. Les éléments sur fond bleu concernent le site général du projet. Ainsi, pour accéder au menu de l'interface il suffit de cliquer sur "The tabletop web interface" depuis le site web de AIDE ^a. Le fichier `style.css` ^b (dans le cercle bleu attaché au menu) correspond au fichier chargé de la mise en forme du site web de AIDE. Il est indépendant du fichier `style.css` propre à l'interface ^c. Les flèches noires représentent les interactions avec le menu. Les flèches violettes représentent les interactions entre les fichiers de l'interface lorsque le jeu est débuté. Les étoiles oranges représentent une liste de fichiers (cités dans la légende) liés aux différentes salles. Ces fichiers sont présentés dans le tableau de la figure 3. Le fichier `client.js` ^d n'est pas présent dans ce tableau car il n'est pas propre à l'interface et a un rôle transversal au sein du projet. Les petits cercles de couleur représentent les interactions avec les fichiers `.js` propres à chaque salle. Par exemple, le petit cercle vert montre que le fichier `room1_1.js` intervient dans le fichier `room1_1.html` au sein de la salle 1. La partie "Bonus" n'est pas encore réalisée à ce jour. Il sera possible d'y voir la suite des aventures des personnages du scénario. Elle est totalement distincte de l'étude menée ici.

a. <http://aide-line.inria.fr>

b. <https://gitlab.inria.fr/line/aide-group/aide/-/blob/master/etc/www/style.css>

c. <https://gitlab.inria.fr/line/aide-group/aide/-/blob/master/@lola-denet/style.css>

d. <https://gitlab.inria.fr/line/aide-group/aide/-/blob/master/src/client.js>

2 Résultats et discussion

La figure 5 montre une capture d'écran du résultat obtenu pour la salle 0 avec la première division du questionnaire affichée.

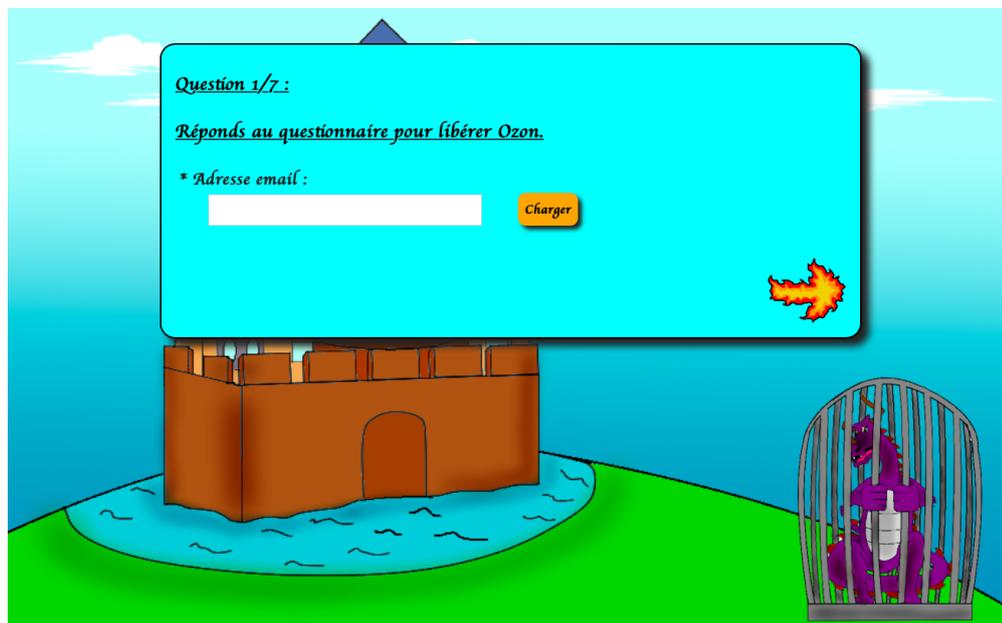


FIGURE 5 – Capture d'écran de la salle 0.

Cette figure montre l'affichage de la salle 0 lorsque l'apprenant clique sur la cage de Ozon. La première division du questionnaire est alors visible et comporte le bouton charger. Le bouton permettant de passer le questionnaire est alors invisible puisqu'il n'y a pas d'adresse saisie. La cage de Ozon ainsi que la flèche de feu sont des images cliquables. Le texte affiché répond à l'utilisation de la fonction `fillString()` et du fichier `string.js`.

La salle 0 est fonctionnelle et a le comportement attendu. Les données saisies sont sauvées dans des fichiers `.json`. Le préremplissage du questionnaire est lui aussi fonctionnel par le bouton "charger" ou par l'url. D'autre part, la vidéo tutorielle apparaît et se lance correctement dans la partie 2 de la salle.

La figure 6 montre l'affichage de la salle 1 avant l'affichage du canevas.

Les salles 1 à 4 ont la même conception. Les graphismes changent en fonction du thème de la salle mais les boutons restent les mêmes pour garder l'homogénéité et la cohérence. Seul le bouton d'entraînement n'est pas présent dans les salles 2 à 4.

Les canevas sont fonctionnels mais il faudra cependant les tester à nouveau lorsque la communication avec la table de jeu sera opérationnelle et les fonctions provisoires réimplémentées.

Il sera également nécessaire de terminer le scénario (texte et audio) sur les parties aide, indices, historique et compétences acquises afin de les ajouter aux pages web.

En l'état actuel, l'apprenant peut accéder à tous les indices quand il le souhaite. Les divisions correspondant à l'aide et aux indices pourront être modifiées de sorte que le nombre d'indices accessibles dépendent du temps passé dans l'activité ou du nombre d'essais ou de l'avancement de l'apprenant.

De plus, il sera nécessaire de réévaluer si les savoirs transmis à la fin de l'activité varient en fonction de ce que l'apprenant a réussi à faire ou non.

D'autre part, il sera peut-être nécessaire de prévoir la possibilité de passer à l'activité suivante



FIGURE 6 – Capture d'écran de la salle 1.

Cette figure montre une capture d'écran de la salle 1. La flèche de feu à gauche permet de revenir à la salle précédente. Ozon est une image cliquable qui permet de répéter la consigne audio. Les trois images en haut à droite sont cliquables et permettent d'ouvrir l'aide (premier bouton en haut avec la tête de Saturnin), d'afficher les consignes au format texte (deuxième bouton en haut avec le parchemin) et de couper/allumer le son (troisième bouton avec le mégaphone magique). Il y a également deux boutons au centre permettant de débiter le jeu (ouvre le canevas et démarre un nouvel audio) ou de faire une séance d'entraînement sur le simulateur.

si l'apprenant ne réussit pas à terminer celle en cours afin que ce ne soit pas trop punitif.

Le comptage des clics devra aussi être implémenté de manière à pouvoir analyser les besoins et le comportement de l'apprenant face aux activités. Il serait peut-être intéressant de pouvoir stocker ces valeurs dans les fichiers où sont gardées les données du questionnaire afin de pouvoir étudier si les réponses aux questions de départ ont un lien avec le comportement de l'apprenant lors des activités d'apprentissage.

Le canevas de la salle 4 affiche les configurations des CréaCubes en vue du dessus. A terme, il serait intéressant de faire en sorte que la visualisation soit en 3 dimensions afin de pouvoir modéliser à l'écran toutes les configurations réalisées par l'apprenant. En effet, en l'état actuel, il y a une perte d'information puisque si deux cubes sont superposés, seul celui du dessus est représenté. Ces modifications pourront être envisagées lorsque la reconnaissance des configurations par apprentissage machine sera optimale.

La figure 7 montre une capture d'écran du simulateur de la salle 1.

La salle 1 est la seule à bénéficier d'un simulateur afin que l'apprenant se familiarise avec le fait de créer un algorithme. Lors du développement de ce simulateur, il était initialement prévu de mettre un menu déroulant avec les valeurs 1, 2 et 3 associé à un seul bouton "avance". Cependant, lors des tests, il s'est avéré que la solution présentée dans la figure 7 était beaucoup plus ergonomique pour un écran tactile.

Enfin, il pourrait être intéressant, à terme, de créer une interface entièrement jouable en ligne



FIGURE 7 – Capture d'écran du simulateur.

Cette capture d'écran montre le canevas du simulateur ainsi que les boutons qui permettent de faire bouger le robot. Ce dernier est représenté par sa photo vue du dessus (en bas à droite du canevas). Les rectangles beiges dessinés sur l'image de fond du canevas sont tracés par-dessus cette dernière afin de représenter les portes. La porte de gauche est placée aléatoirement à la position A ou B. La porte en haut à droite représente la sortie fermée. Le rectangle pivote pour simuler l'ouverture de la porte lorsque le robot s'arrête sur la case contenant la clé. Le tableau sur la gauche représente l'endroit où les plaques d'instructions s'affichent lorsque l'apprenant débute le jeu. Les boutons oranges à droite sont les boutons permettant de mouvoir le robot. Lorsque ce dernier pivote, il le fait sur place et ce sont les boutons "Avance" qui lui permette de changer de case. Le robot devient invisible lorsqu'il passe sous le passage secret et revient au départ s'il tombe dans le trou noir (piège). Lorsqu'un mouvement le fait rencontrer un mur ou une porte close, une notification apparaît et le robot est ramené sur la case où le mouvement a débuté. L'entraînement se termine lorsque le robot est arrivé devant la porte de sortie et que celle-ci est ouverte.

afin de comparer les résultats obtenus par les apprenants sur les activités de manière hybride (en partie avec des outils débranchés et en partie avec des outils connectés, comme ici) ou de manière totalement connectée (sans tables de jeu).

Conclusion

Pour conclure, il semble important de rappeler que ce projet inclut plusieurs disciplines : les neurosciences cognitives, les sciences informatiques et les sciences de l'éducation. Cette alliance permet de travailler sur une problématique aux multiples enjeux tant sur le plan de l'informatique, de l'éducation que sur l'environnement et la société. Elle met en avant l'importance de connaître les origines des outils numériques utilisés au quotidien afin de profiter pleinement de leur potentiel sans qu'ils ne deviennent délétères. De plus, l'étude liée à ce projet pourrait permettre de développer de nouvelles méthodes d'enseignement ou du moins d'adapter les méthodes actuelles afin de, par exemple, limiter le décrochage scolaire. L'étude des mécanismes d'apprentissage par la combinaison des neurosciences et des sciences de l'éducation pourrait permettre d'en améliorer la compréhension. Le développement de la pensée computationnelle contribuera grandement à cela.

Références

- [Barnabé *et al.*, 2020] BARNABÉ, S., DENET, L., MANRIQUE, M., MENON, D., PASCUAL, , ROMERO, M. et VIÉVILLE, T. (2020). A low-cost tabletop set-up to collect learning analytics during computational thinking unplugged or tangible activities. Rapport technique, Inria. to appear.
- [Lachaux, 2018] LACHAUX, J.-P. (2018). Éduquer la métacognition, la clé du succès pour les enfants! *Cerveau et psycho*, 105:82–84.
- [Menon *et al.*, 2019] MENON, D., ROMERO, M. et VIÉVILLE, T. (2019). Computational thinking development and assessment through tabletop escape games. *International Journal of Serious Games*, 6(4):3–18.
- [Romero *et al.*, 2018] ROMERO, M., NOIRPOUDRE, S. et VIÉVILLE, T. (2018). Que disent les sciences de l'éducation à propos de l'apprentissage du code? *Revue de l'EPI (Enseignement Public et Informatique)*.



**RESEARCH CENTRE
BORDEAUX – SUD-OUEST**

200 avenue de la Vieille Tour
33405 Talence Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-0803