CRITIQUE DU LANGAGE-AUTEUR "AUTHORWARE"

Pierre DILLENBOURG, François LOMBARD

Cet article présente quelques points forts et points faibles du langage-auteur Authorware Professional™, produit par la société Macromedia. Il s'agit d'un langage de haut niveau spécialisé pour la conception de logiciels éducatifs, dans la lignée des langages Tutor, Pilot et autre TenCORE™. Authorware est plus spécialisé que des systèmes tels que Hypercard™, Visual Basic™ ou Toolbook™ qui n'ont pas été conçus spécifiquement pour développer des applications éducatives. Il se différencie aussi de produits tels que Director™ qui permettent de créer des présentations : ces derniers sont plus performants sur le plan des effets visuels et sonores (surtout les animations 3D), mais sont moins riches sur le plan de l'interaction. Authorware fonctionne sur Mac et sous Windows et possède une interface auteur qui est pratiquement identique sur les deux machines.

Un langage-auteur doit concilier trois caractéristiques pourtant partiellement contradictoires : la convivialité, la productivité et la puissance. Nous illustrons ces trois caractéristiques par différents aspects d'Authorware.

La **facilité d'apprentissage et d'utilisation ou convivialité** est la caractéristique première d'un langage-auteur, puisque celui-ci s'adresse à un public d'enseignants et non d'informaticiens. Sur cet aspect, Authorware réalise son meilleur score. L'auteur construit son programme en agençant des icônes sur l'écran. Les icônes sont connectées par des flèches qui dessinent un algorithme. En double-cliquant sur ces icônes, l'auteur édite certains paramètres. Nous présentons les icônes de base :

- Dans l'icône de présentation. L'auteur dessine un écran tel qu'il apparaîtra à l'élève au moyen d'une petite palette graphique (ou importe des images ou dessins créés dans un autre logiciel). Le contenu d'une icône se superpose aux contenus des icônes précédentes comme des transparents que l'on superpose sur un rétroprojecteur.

- L'icône d'effacement efface le contenu des icônes de présentation auxquelles elle est associée. Il n'est pas possible d'effacer (ou de déplacer) séparément les objets d'une même icône de présentation. Aussi, si deux objets doivent être effacés à différents moments, ils devront être dessinés dans deux icônes différentes.
- L'icône d'interaction présente un stimulus (une question, une liste de choix,..), définit un certain nombre de réponses, associe un feed-back à chaque réponse, prévoit le nombre d'essais ou le temps maximal (total ou par essai) et spécifie la modalité de réponse : bouton, texte, menu, zone sensible, objet sensible, objet à déplacer.
- Les icônes multimédia permettent d'insérer un son, une mélodie, une image, ou un film, ainsi que de piloter un magnétoscope.
- L'icône d'animation, malgré son nom, permet uniquement de déplacer le contenu d'une icône de présentation (soit l'ensemble des objets de cette icône). Elle dispose néanmoins de plusieurs modalités de déplacement, dont la vitesse et la destination peuvent être fixées par des variables créées par l'auteur.
- L'icône de décision permet de créer des structures conditionnelles et itératives.
- L'icône de calcul permet d'effectuer des opérations sur des variables (par exemple, le pourcentage de réponses correctes) et d'utiliser un jeu de fonctions disponibles dans Authorware (par exemple pour dessiner des formes, enregistrer des données, calculer le délai entre deux 'click'...).

L'utilisation et l'édition de ces icônes ne pose guère de difficulté. Depuis trois ans, nous enseignons ce langage à des étudiants et à des enseignants qui ont peu de compétences en informatique. Toutefois, pour développer des logiciels d'une certaine complexité, l'auteur est tôt ou tard confronté au concept de variable. Ce concept s'avère plus difficile pour certains.

La **productivité** d'un langage-auteur est la rapidité avec laquelle un auteur crée et modifie un programme. Sur ce plan également, Authorware dispose d'importants atouts. Comme tout langage-auteur digne de ce nom, Authorware comporte des procédures puissantes d'analyse de réponse de type 'texte'. Celles-ci permettent notamment de définir des tolérances orthographiques, de définir des contraintes sur l'ordre des mots, la ponctuation ou les majuscules, ainsi que d'accepter que les différents éléments d'une réponse soient fournis en plusieurs fois. Il s'agit de techniques de 'pattern matching' simple qui identifient un ou

plusieurs mots dans la réponse de l'élève, mais ne permettent pas de réaliser des analyses syntaxiques complexes telles qu'elles existent en intelligence artificielle.

Un autre facteur de productivité d'Authorware est la facilité avec laquelle on peut passer du mode 'auteur' (édition du programme) au mode 'élève' (exécution du programme) et vice-versa. Lorsque le didacticiel en construction est exécuté (mode élève), l'auteur peut à tout moment interrompre le programme, modifier un élément et reprendre l'exécution au point où elle avait été interrompue. Cette fonctionnalité représente un gain de temps très important. Dans d'autres langages, il faut quitter le mode 'élève', éditer le code et relancer le programme dès le départ. Dans Authorware, l'auteur peut en outre demander d'exécuter le programme à partir d'un point donné (en positionnant un drapeau dans l'algorithme) afin de tester un morceau particulier du programme.

Le travail du développeur est aussi réduit par le fait qu'Authorware gère automatiquement une centaine de variables prédéfinies, telles que le pourcentage de réponses correctes fournies depuis le début du didacticiel, le nombre d'essais à une question donnée, le temps de latence pour chaque réponse, le nombre d'éléments corrects ou incorrects dans une réponse à éléments multiples, la position du sujet dans le cours, etc. Ces variables couvrent l'essentiel des besoins les plus fréquents d'un concepteur de didacticiels.

Le point faible d'Authorware en matière de productivité est la lenteur de programme, en particulier lors du traitement des icônes de calcul. Certes, cette lenteur varie selon la machine. Néanmoins, les auteurs les plus avancés en informatique ont tendance, lorsque le programme comprend des calculs complexes, à réaliser ces calculs dans un autre langage (C, Pascal...) et à appeler ces fonctions extérieures depuis Authorware au moyen d'une XCMD ou DLL.

Un autre point faible est lié à l'interface de programmation graphique : la difficulté d'effectuer des changements globaux sur le programme. Par exemple, imaginons que l'auteur crée 5 zones de réponses pour lesquelles il exige que le sujet clique une fois. Il change ensuite d'avis et décide que le sujet devra cliquer deux fois. Il devra dans ce cas rééditer manuellement chaque réponse, afin de modifier le nombre de 'click' requis. Par contre, il est maintenant possible d'effectuer des changements globaux sur les textes présentés dans les icônes.

Vu la gourmandise mnémonique des didacticiels multimédia, Authorware offre la possibilité d'associer une bibliothèque d'images ou de sons à un didacticiel. Ainsi, une image complexe qui doit être utilisée

plusieurs fois dans le cours sera stockée une seule fois dans le programme. Par contre, Authorware ne permet pas (encore) de créer de procédures. L'auteur peut utiliser un certain nombre de fonctions prédéfinies, mais il ne peut définir ses propres fonctions. Il existe certes quelques astuces pour remédier localement à cette lacune, mais sans atteindre la productivité de véritable procédures, en particulier la récursivité.

Authorware exploite enfin le concept de modèle : il s'agit d'une partie de programme réutilisable dans un autre programme. Cette option ne s'éloigne pas fondamentalement du couper-coller : l'auteur pourrait simplement sélectionner un morceau de didacticiel existant et le coller dans le didacticiel en construction. L'avantage du 'modèle' est qu'au moment de l'insertion du modèle dans le nouveau programme, Authorware gère les conflits éventuels entre les variables existant déjà dans le didacticiel en construction et celle importées dans le modèle.

Enfin, dernier facteur de productivité, Authorware dispose d'une passerelle Mac - PC : un programme développé sur Mac peut être transféré sur PC et vice-versa. Le transfert fonctionne relativement bien, mais soulève quelques problèmes pour la traduction des polices de caractères ainsi que pour le stockage des images.

La **puissance** d'Authorware est certainement son point le plus faible. En règle générale, la productivité et la convivialité sont obtenues au détriment de la puissance : la création est plus facile et plus rapide, mais les didacticiels créés sont prisonniers d'un canevas prédéfini. Par son interface graphique, Authorware favorise une conception assez classique des didacticiels. La structure de base est une séquence de questions et de réponses. A chaque réponse, l'auteur peut associer un sous-programme. Celui-ci contiendra par exemple une autre séquence et d'autres questions. Les sous-programmes s'emboîtent les uns dans les autres. Ces boîtes dans des boîtes forment une structure arborescente. Authorware favorise donc une approche modulaire dans laquelle le curriculum est fortement structuré. La notion clé est la séquence temporelle des icônes.

Authorware 3 offre entre autre des structures de graphe (par exemple, des hypertextes), qui présupposent des connexions directes entre différentes feuilles d'un arbre.

Authorware s'avère moins pertinent pour les didacticiels de type micro-monde dans lequel le sujet réagit face à une situation complexe. Ce langage-auteur offre toutefois une fonctionnalité très intéressante pour ce type de didacticiels : la possibilité de définir des réponses dites

'perpétuelles'. Un menu par exemple est une sollicitation implicite à laquelle le sujet peut répondre n'importe quand. Les réponses perpétuelles peuvent être des boutons, des objets sensibles, etc. Toutefois, ce genre de logiciels implique souvent des calculs importants, qui s'avèrent assez lents avec Authorware.

En d'autres termes, Authorware a un biais vers les didacticiels de type 'question-réponse'. Ce biais peut être perçu à la fois comme positif par rapport aux techniques qui se limitent à la présentation d'informations, mais comme négatif par rapport aux méthodes qui privilégient l'exploration active de l'élève. Ce caractère un peu fermé du système est dû au fait qu'il n'existe pas 'derrière' Authorware de langage de programmation qui soit accessible à l'auteur. Authorware ne produit pas un code que l'auteur pourrait ensuite éditer 'à la main'. En dehors de icônes de calcul, un didacticiel créé avec Authorware ne comporte aucune ligne de code. Authorware se différencie donc nettement des langages tels qu'Hypercard ou Toolbook, dans lequel un informaticien conservateur se sentira probablement plus à l'aise. Il est par exemple inutile d'imprimer un programme Authorware : cette impression ne permet pas une meilleure lecture que la présentation du programme à l'écran.

Un langage est plus puissant s'il permet de définir des structures abstraites. On pourrait penser qu'un langage reposant sur l'agencement d'icônes ne permet aucune abstraction. En réalité, bien qu'il s'agisse d'un langage de programmation graphique, Authorware permet une abstraction remarquable. En effet, lorsque l'auteur édite une icône, il doit fournir certaines valeurs : nombre d'essais, temps maximum, position ou vitesse d'une animation, nombre d'itérations, dimensions d'une zone de réponse.... La plupart de ces valeurs peuvent être remplacées par des variables. Il est ainsi possible de créer des structures génériques et de les instancier en cours d'exécution. Ces fonctionnalités sont en particulier nécessaires pour créer des interaction adaptées au niveau de l'apprenant. Authorware dispose d'un générateur de nombre aléatoires comme la plupart des langages de programmation. Les paramètres peuvent également être stockés dans un vecteur de variables. Authorware ne dispose par contre pas de tables de variables à deux ou plusieurs dimensions.

La possibilité d'abstraction concerne même le design des écrans. Au lieu de poser un objet graphique, un bouton ou une zone texte à un point précis de l'écran, l'auteur peut définir la position de ces objets au moyen de variables. Tout texte affiché à l'écran peut soit être entré tel quel par l'auteur, soit être contenu dans une variable. Certaines formes

géométriques simples peuvent être dessinées au moyen de fonctions (icône de calcul).

Enfin, autre forme d'abstraction, Authorware représente les structures d'interactions indépendamment de la modalité de réponse. Par exemple, pour la question "Qui a gagné la bataille de Waterloo", l'auteur pourra prévoir trois réponses : Wellington, Blücher et Napoléon. Pour chaque réponse, il fournira un feedback spécifique. Après une réponse erronée, il invitera l'élève à fournir une nouvelle réponse. Dans Authorware, cette structure restera identique quelle que soit la modalité de réponse : l'élève peut entrer les noms au clavier, cliquer sur des boutons correspondant à chaque nom, cliquer sur un portrait des protagonistes.

CONCLUSIONS

Authorware réalise un compromis que nous considérons comme remarquable, car il atteint une grande facilité et productivité sans sacrifier totalement à la puissance. Ses producteurs le qualifient d'outil professionnel et on peut penser qu'il mérite ce titre. Son prix mérite malheureusement également ce label. Authorware est très efficace au sein de la niche spécifique des logiciels à visée éducative et hautement interactifs. Il ne s'agit pas d'un outil de production multimédia au sens large. La philosophie des constructeurs est que les composantes multimédia (son, images fixes ou animées) doivent être créées dans des logiciels spécialisés et assemblées dans Authorware, au sein d'un scénario interactif. Ses points forts sont la facilité de création de scénario et la richesse des formes d'interaction. Nous ne recommandons pas Authorware pour le développement de produits faiblement interactifs, dont le déroulement est linéaire. Nous le recommandons pour la construction de didacticiels centrés sur les activités de l'apprenant.

Cet article repose sur notre expérience avec Authorware 2. La version 3 est aujourd'hui disponible.

Pierre DILLENBOURG [1], François LOMBARD [2]

1 TECFA, Unité de Technologie de Formation et Apprentissage, Faculté de Psychologie et des Sciences de l'Education, Université de Genève, 9 Route de Drize CH1227 Carouge, Suisse. pdillen@divsun.unige.ch.

2 Centre Informatique Pédagogique, Case Postale 3144, 1211 Genève 3, Suisse. lombard@cui.unige.ch.