aLDEAS : un langage de définition de systèmes d'assistance épiphytes (Prix du meilleur article d'IC2014)

Blandine Ginon, Stéphanie Jean-Daubias, Pierre-Antoine Champin, Marie Lefevre

aLDEAS : un langage de définition de systèmes d'assistance épiphytes

Blandine Ginon[1,2], Stéphanie Jean-Daubias[1,3], Pierre-Antoine Champin[1,3] et Marie Lefevre[1,3]

[1] Université de Lyon, CNRS,
[2] INSA-Lyon, LIRIS, UMR5205, F-69621, France
[3] Université Lyon 1, LIRIS, UMR5205, F-69622, France
{prenom.nom}@liris.cnrs.fr

Résumé : Nous proposons un langage qui permet de spécifier des systèmes d'assistance pour une application-cible donnée, sous la forme d'un ensemble de règles. Ce langage est complété par plusieurs patrons d'actions d'assistance. Nous avons mis en œuvre ces propositions à travers un éditeur d'assistance à destination des concepteurs d'assistance et un moteur générique d'assistance permettant d'exécuter l'assistance spécifiée pour les utilisateurs finaux de l'application-cible sans avoir à la modifier.

Mots-clés : assistance à l'utilisateur, systèmes épiphytes, langage.

1 Introduction

L'assistance aux utilisateurs est l'une des solutions pour pallier les difficultés de prise en main et d'utilisation des applications informatiques. De tels dispositifs permettent d'éviter la sous-exploitation ou le rejet du logiciel.

Nous définissons l'assistance comme l'ensemble des moyens qui permettent de faciliter la prise en main et l'utilisation d'une application, de manière adaptée à l'utilisateur et au contexte d'utilisation. L'assistance vise à permettre à l'utilisateur d'exploiter pleinement toutes les possibilités d'une application, et elle facilite l'appropriation des connaissances et compétences nécessaires à l'utilisation de cette application. Elle comprend les quatre types d'assistance définis par (Gapenne et al., 2002) : substitution, suppléance, assistance et aide.

Le développement d'un système d'assistance adapté à une application est une tâche complexe et coûteuse, souvent négligée par les concepteurs d'applications informatiques. Une personne autre que le concepteur de l'application peut alors souhaiter adjoindre un système d'assistance à une application qui en est dépourvue, ou qui possède un système d'assistance incomplet. Par exemple, dans le cadre d'une communauté d'utilisateurs, un expert peut souhaiter concevoir un système d'assistance pour faire bénéficier de son expérience des utilisateurs plus novices. Le code source de l'application est la plupart du temps non disponible dans le cas où le concepteur de l'assistance n'est pas le concepteur de l'application-cible ; il n'est alors pas possible d'intégrer directement un système d'assistance dans l'application. De plus, comme dans notre exemple, le concepteur potentiel de l'assistance n'est pas toujours un programmeur. Une alternative à l'approche classique de développement d'un module d'assistance intégré dans une application consiste à adopter une démarche épiphyte pour permettre *a posteriori* la spécification et l'exécution d'un système d'assistance dans une application existante sans avoir à la modifier. Un assistant épiphyte est un assistant capable de réaliser une action dans une application-cible externe, sans perturber son fonctionnement (Paquette, et al., 1996).

Le travail présenté dans cet article se situe dans le contexte du projet AGATE (Approche Générique d'Assistance aux Tâches complexEs), qui vise à proposer des modèles génériques et des outils unifiés pour permettre la mise en place de systèmes d'assistance dans des applications existantes, que nous appelons applications-cibles. Nous avons proposé pour cela un processus d'adjonction d'un système d'assistance à une application-cible en deux phases, cf. Figure 1 et (Ginon, et al., 2013b). La première phase concerne un expert de l'application-

cible, appelé par la suite concepteur de l'assistance. Elle permet au concepteur de spécifier l'assistance qu'il souhaite pour une application-cible, en définissant un ensemble de règles d'assistance. La seconde phase concerne les utilisateurs finaux de l'application-cible. Elle consiste en l'exécution de l'assistance souhaitée par le concepteur. Cette phase a lieu à chaque utilisation de l'application-cible par un utilisateur, et se compose de trois processus. La surveillance de l'application-cible (Ginon, et al., 2013a) permet d'observer en continu et de tracer toutes les interactions entre l'utilisateur et l'interface de l'application-cible. En parallèle, le processus d'identification d'un besoin d'assistance exploite les règles d'assistance définies par le concepteur et déclenche le processus d'élaboration d'une réponse au besoin identifié. La réponse se fait sous la forme d'une action d'assistance, réalisée dans l'application-cible par un assistant épiphyte.

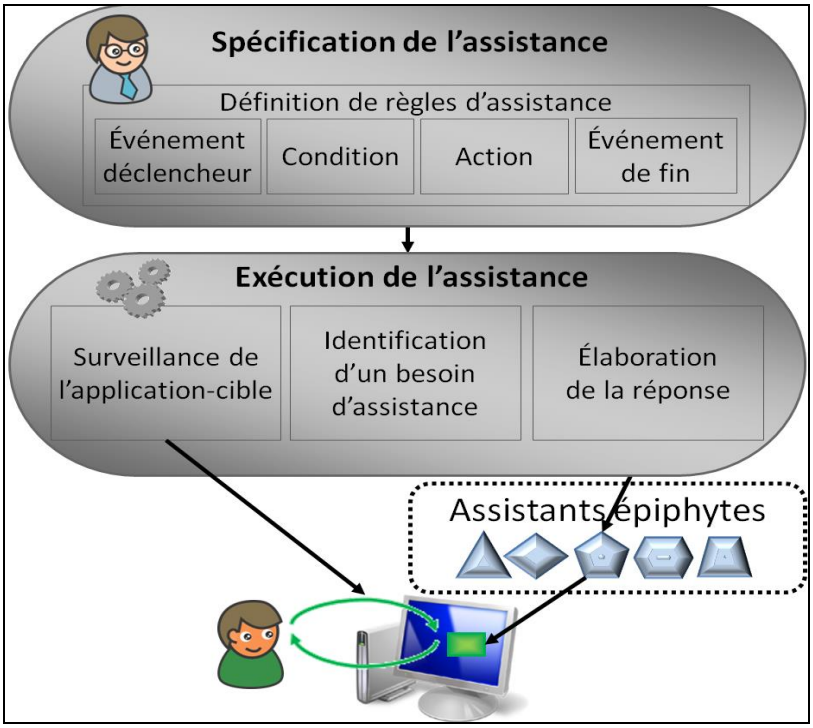

FIGURE 1 – Processus d'adjonction d'un système d'assistance à une application existante.

Dans cet article, nous détaillons aLDEAS, le langage de définition de systèmes d'assistance que nous avons défini afin de permettre la spécification de systèmes d'assistance selon cette approche. Ce langage nous a notamment permis de définir des patrons d'actions d'assistance composées présentés en section 4. Nous présentons ensuite la mise en œuvre de ces patrons dans le système SEPIA. Enfin, nous exposons les évaluations que nous avons effectuées pour ces propositions.

2 État de l'art

La spécification *a posteriori* de systèmes d'assistance pour des applications-cibles existantes a fait l'objet de plusieurs travaux. Les approches de (Paquette, 2012) et (Dufresne, et al., 2003) permettent d'ajouter un système conseiller à un scénario de l'environnement Telos pour le premier et de l'environnement ExploraGraph (Dufresne, 2001) pour le second. Ces systèmes conseillers sont définis par un ensemble de règles de la forme <événement déclencheur, condition de déclenchement, action d'assistance, événement de fin>. Les conditions de déclenchement peuvent inclure une consultation du profil de l'assistance et de

l'historique de l'assistance afin de personnaliser et contextualiser l'assistance. Les actions d'assistance proposées sont de type *message textuel* affiché dans une fenêtre pop-up pour Telos et *animation* ou *message* transmis par un agent animé pour ExploraGraph. L'approche proposée par (Richard, et al., 2004) permet l'ajout d'un système conseiller dans une application Web, afin de permettre le déclenchement d'actions d'assistance lors d'un clic sur un lien. Les actions proposées sont de type *messages textuels* affichés dans une fenêtre popup, les messages peuvent contenir des liens vers une page web ou vers des ressources liées à l'assistance. L'assistance peut être personnalisée en fonction d'un historique de la navigation. Le modèle CAMELEON (Carlier, et al., 2010) permet quant à lui d'ajouter un agent animé capable de se déplacer, de réaliser des *animations* et d'afficher des *messages* dans une application web, grâce à des balises insérées de manière épiphyte dans la page web.

Ces différentes approches ne peuvent cependant pas être utilisées dans n'importe quelle application. Elles sont en effet spécifiques à un environnement donné ou au web. Dans le cadre du projet AGATE, nous nous sommes intéressés à l'adjonction *a posteriori* de systèmes d'assistance dans des applications existantes les plus variées, sans que ces applications soient spécifiques à un domaine ou à un environnement.

3 aLDEAS : un langage de définition de systèmes d'assistance

Pour éviter les confusions avec les styles standards de Word (notamment lors des fusions de documents), les styles spécifiques à la conférence IC2012 sont précédés du préfixe « aic ». En voici la liste et l'usage :

Nous proposons aLDEAS (a Language to Define Epi-Assistance Systems), un langage opérationnalisé dans un outil à destination des concepteurs d'assistance, dont le but est de permettre la définition de systèmes d'assistance sous la forme d'un ensemble de règles, pour des applications existantes. Ce langage est constitué de différents types de composants (cf. Figure 2) que nous présentons de façon détaillée dans cette section. Nous montrerons par la suite comment ces composants peuvent être combinés pour créer des actions d'assistance, qui répondront à des besoins d'assistance.

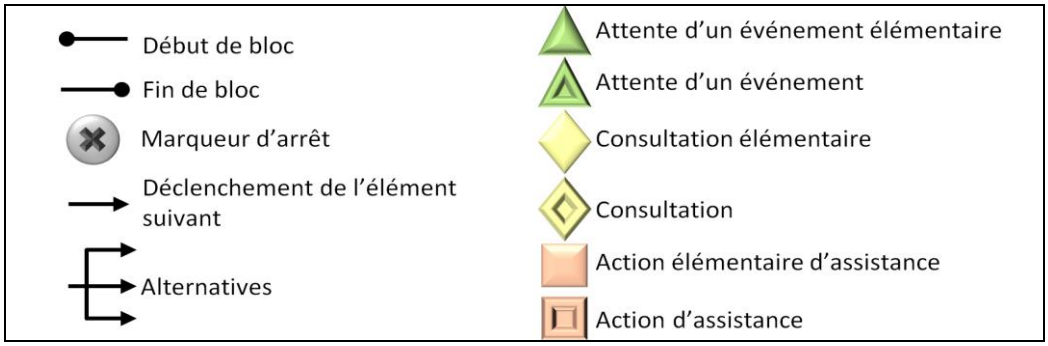

FIGURE 2 – Composants du langage aLDEAS.

3.1 Les attentes d'événements

Notre langage propose des éléments qui permettent d'attendre qu'un événement donné ait eu lieu avant de déclencher un autre élément, telle qu'une action d'assistance. Les attentes d'événements élémentaires, représentées dans le langage par  , peuvent être liées à une **action de l'utilisateur**, comme un clic sur un bouton donné, elles peuvent concerner une **action du système d'assistance**, comme le déclenchement ou la fin d'une règle d'assistance, ou la **fin d'un timer**. Un timer est associé à une durée, et peut être déclenché à la suite de n'importe quel événement lié à l'action de l'utilisateur ou à l'assistance. Par exemple, un

timer peut être utilisé pour spécifier qu'une action d'assistance durera 30 secondes. Un timer peut également être utilisé pour définir d'une action d'assistance sera déclenchée si l'utilisateur reste inactif pendant 5 minutes.

Les attentes d'événements élémentaires peuvent être combinées pour former des attentes d'événements composées, représentées par . Ainsi, ces éléments permettent d'attendre une succession donnée d'événements élémentaires formant un événement de plus haut niveau, par exemple une action de correction des yeux rouges sur une photo.

3.2 Les consultations

Le langage aLDEAS propose plusieurs types de consultations élémentaires, représentées dans le langage par , qui peuvent être utilisées pour personnaliser et contextualiser l'assistance. aLDEAS permet la consultation directe de l'utilisateur, pour lui permettre de choisir entre plusieurs options par exemple. il permet également la consultation de l'état de l'application-cible, afin de connaître le texte d'une zone de saisie, ou de savoir quel item est sélectionné dans une liste déroulante par exemple. Enfin, aLDEAS permet la consultation des ressources liées à l'assistance, comme le profil de l'utilisateur qui contient notamment des informations sur les préférences de l'utilisateur en matière d'assistance, l'historique de l'assistance qui contient des informations sur les règles et actions déclenchées pour un utilisateur, et les traces de l'utilisateur qui contiennent des informations sur toutes les interactions entre l'utilisateur et l'interface de l'application-cible.

Les consultations élémentaires peuvent être combinées par une formule logique afin de créer une consultation composée, représentée par . Les consultations, élémentaires ou composées, renvoient une valeur dont le type varie : booléen, texte, ou nombre. Par exemple, une consultation de l'historique de l'assistance peut renvoyer un nombre qui indique combien de fois une action d'assistance a été déclenchée.

3.3 Les actions élémentaires d'assistance

Notre langage propose deux catégories d'actions élémentaires, représentées par : des actions intégrées et des actions extérieures à l'interface de l'application-cible.

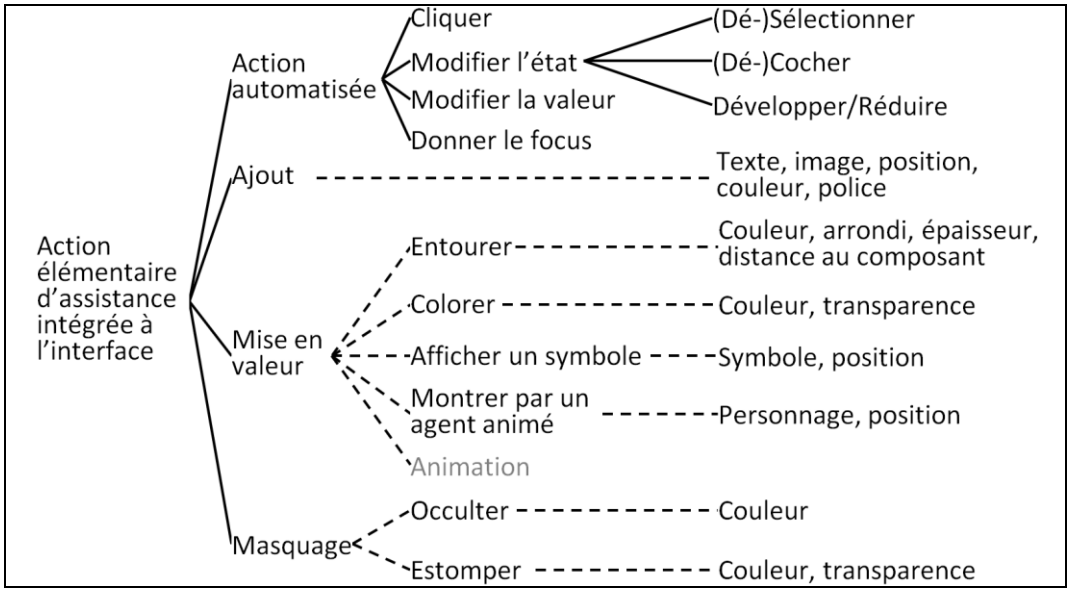

FIGURE 3 – Actions élémentaires d'assistance intégrées à l'interface de l'application-cible.

Les actions intégrées agissent directement sur l'interface de l'application-cible et concernent un composant donné, comme un bouton ou une zone de saisie. Le langage propose quatre types d'actions sur un composant (cf. Figure 3) : **action automatisée**, pour agir à la place de l'utilisateur ; **ajout de composant**, pour enrichir l'interface de l'application-cible, par exemple un bouton permettant de demander de l'aide ; **mise en valeur**, pour guider l'utilisateur et attirer son attention sur un composant ; et **masquage**, pour simplifier aux yeux de l'utilisateur l'interface de l'application-cible. Une action intégrée à l'interface de l'application-cible peut être associée à plusieurs paramètres optionnels, indiqués sur la Figure 3 par des traits en pointillés. Par exemple, pour une mise en valeur, un composant de l'interface de l'application-cible peut être désigné par un agent animé ou entouré par un trait d'une couleur et d'une épaisseur données.

Les actions extérieures à l'interface permettent de proposer à l'utilisateur de l'assistance non associée à un composant de l'interface de l'application-cible. Notre langage en propose trois types (cf. Figure 4) : les **messages langagiers**, associés à un texte pouvant être affiché et/ou lu; les **animations**, par exemple un agent animé qui applaudit; et les **ressources** qui peuvent être proposées à l'utilisateur, par exemple une vidéo de démonstration, un forum ou une application comme la calculatrice.

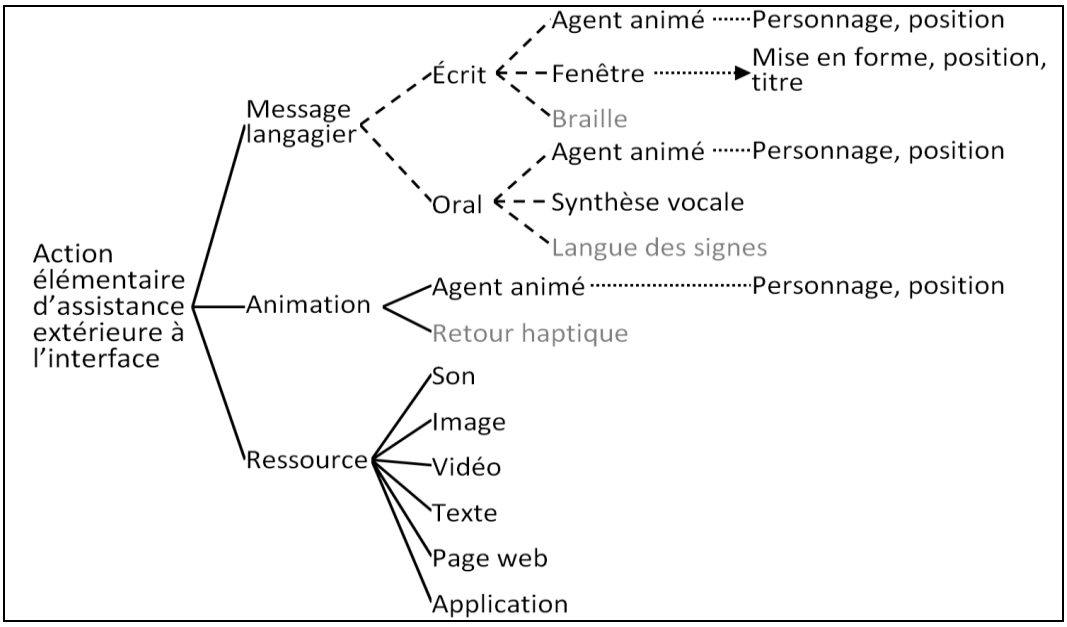

FIGURE 4 – Actions élémentaires d'assistance extérieures à l'interface de l'application-cible.

3.4 Définition d'actions d'assistance composées

Tous les éléments proposés par le langage aLDEAS peuvent être combinés pour créer des actions d'assistance composées, représentées dans le langage par . Par exemple, une action peut être composée d'une séquence d'actions élémentaires ou d'une action associée à un événement de fin, c'est-à-dire une attente d'événement suivie du marqueur qui provoque la fin de toutes les actions lancées depuis le marqueur de début et non encore terminées. La Fig. 5 donne l'exemple d'une action contenant deux actions élémentaires d'assistance : un message et une mise en valeur. Cette action contient également un événement de fin : le message et la mise en valeur disparaîtront au bout de 30 secondes.

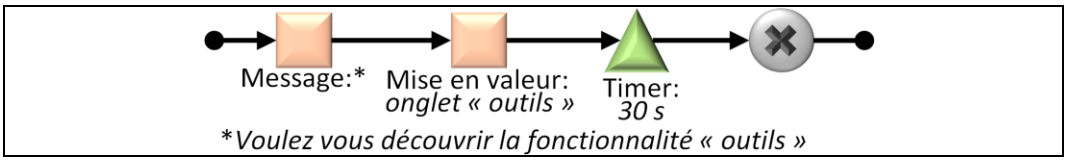

Fig. 5 – Exemple d'une action d'assistance composée.

Dans le cas où une action d'assistance contient plusieurs marqueurs d'arrêt, elle peut renvoyer un état de type texte, indiqué dans le langage sous chaque marqueur d'arrêt (cf. exemple Figure 9 et Figure 10). Les éléments renvoyant une valeur peuvent être suivis d'une alternative à n branches chacune associée à un test dont le type dépend du type de valeur renvoyée. Par exemple pour un texte, les tests peuvent imposer qu'il soit égal, qu'il contienne, commence ou termine par un texte donné. Si un test est vérifié, l'élément suivant de la branche est déclenché. Plusieurs éléments peuvent être déclenchés en parallèle si la valeur de retour vérifie plusieurs tests. Par exemple 4 vérifie $4<7$ et $4 \in [2\ ;\ 9]$. Un exemple d'alternative est donné dans la règle R0 de la Figure 7.

4 Patrons exploitant le langage aLDEAS

Afin de faciliter la définition avec aLDEAS d'actions composées, nous proposons un ensemble de patrons. Pour cela nous enrichissons notre langage avec deux structures supplémentaires : l'embranchement « ou » et les éléments optionnels précédés d'un « ? ». Contrairement aux structures présentées en section 3, celles-ci ne décrivent pas l'exécution de l'assistance ; ce sont des choix à faire par le concepteur de l'assistance au moment de l'instanciation du patron.

4.1 Patron de règles d'assistance

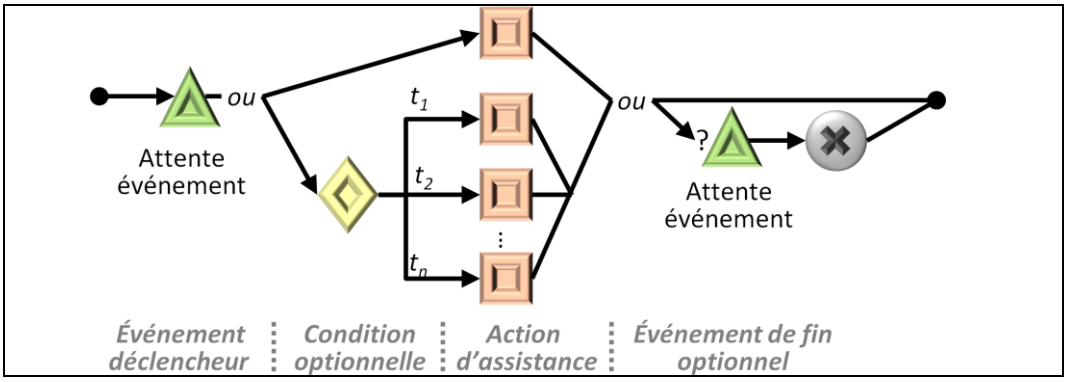

FIGURE 6 – Patron de règles d'assistance.

Le but de notre langage est de permettre à des concepteurs de spécifier par un ensemble de règles l'assistance qu'ils souhaitent pour une application-cible. Dans aLDEAS, nous définissons une règle d'assistance comme une action composée qui respecte le patron donné en Figure 6. Une règle débute par un événement déclencheur qui correspond à l'attente d'un événement. Une règle contient ensuite une action d'assistance ou une consultation avec alternatives associées chacune à une action d'assistance. Une règle peut enfin être associée à un événement de fin, qui correspond à l'attente d'un événement suivi du marqueur d'arrêt. Si une règle n'est pas associée à un événement de fin, elle ne se terminera que lorsque toutes les actions qu'elle a déclenchées se seront achevées. En effet, une action d'assistance peut être associée à son propre événement de fin, et dans certains cas, l'utilisateur peut y mettre fin lui-

même. Par exemple, dans le cas d'un message affiché par une fenêtre pop-up, l'utilisateur peut mettre fin à l'action en supprimant la fenêtre. La Figure 7 donne l'exemple de deux règles d'assistance créées pour l'application-cible PhotoScape1, un logiciel gratuit de retouche d'images. La règle R0 contient un événement déclencheur (le lancement de l'assistance) et une consultation de l'utilisateur lui demandant s'il souhaite de l'aide. R1 sera déclenchée si l'utilisateur choisit l'option « oui, je veux de l'aide ». La règle R7 est déclenchée par le clic de l'utilisateur sur le composant d'identifiant 228 (qui correspond à l'onglet « outils » de PhotoScape). R7 déclenche une action d'assistance constituée de deux actions élémentaires d'assistance : la mise en valeur du composant d'identifiant 210 (le bouton « yeux rouges » de PhotoScape) ; et un message suggérant à l'utilisateur de cliquer sur ce bouton. Le clic de l'utilisateur sur le bouton « yeux rouges » mettra fin à la règle R7 et provoquera donc l'effacement de la mise en valeur et du message.

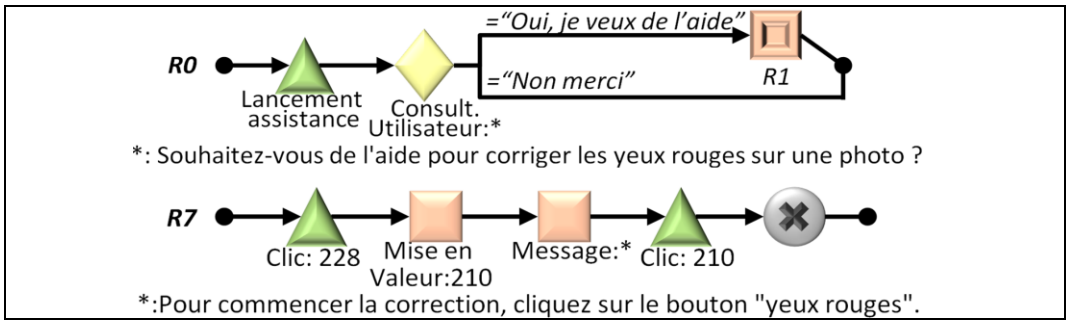

FIGURE 7 – Exemple des règles d'assistance R0 et R7 pour PhotoScape.

4.2 Patron d'actions d'assistance

Le langage aLDEAS permet la définition d'actions d'assistance complexes, combinant de nombreux éléments d'assistance. La définition de telles actions peut être difficile, or aLDEAS s'adresse principalement à des concepteurs d'assistance qui ne sont pas nécessairement informaticiens. Pour cette raison, nous avons défini des patrons d'actions composées, associés à notre langage, afin de faciliter la définition de certaines actions composées fréquemment présentes dans les systèmes d'assistance existants : **actions d'agent animé**, **présentations guidées** et **pas à pas**. Dans cette section, seuls les patrons relatifs aux actions de type pas à pas sont donnés.

Une **action d'agent animé** permet de combiner plusieurs actions élémentaires d'un même personnage : messages, animations (montrer un composant, applaudir, saluer…), et déplacements à l'écran. Par exemple : l'agent animé se place à côté du champ e-mail, il affiche le message « n'oublie-pas de remplir ton adresse mail », il montre le champ e-mail jusqu'à ce que l'utilisateur ait modifié sa valeur.

Une **présentation guidée** comporte plusieurs étapes, dans lesquelles un composant est mis en valeur et éventuellement présenté par un message. On retrouve fréquemment ces actions d'assistance dans les applications existantes, notamment lorsqu'une application est lancée pour la première fois, ou à la suite d'une mise à jour.

Un **pas à pas** vise à faciliter la réalisation d'une tâche en la détaillant sous forme de plusieurs étapes. Chaque étape correspond à une action à réaliser sur un composant de l'interface de l'application-cible. Nous appelons pas à pas automatisé un pas à pas dans lequel le système d'assistance va réaliser les actions à la place de l'utilisateur. Nous appelons pas à pas guidé un pas à pas dans lequel le système va demander à l'utilisateur de réaliser lui-même les actions. Les patrons d'étape de pas à pas automatisé et guidé sont donnés respectivement

[1] http://www.photoscape.org

en Figure 9 et Figure 10 : ces deux patrons d'étapes sont exploités par le patron de pas à pas (cf. Figure 8), sur lequel les étapes sont représentées par .

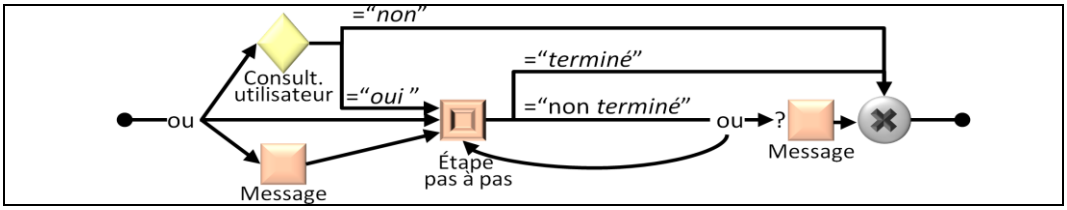

FIGURE 8 – Patron de pas à pas.

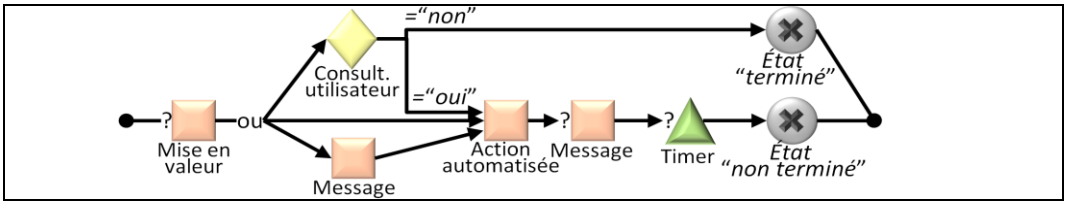

FIGURE 9 – Patron d'étape de pas à pas, en mode automatisé.

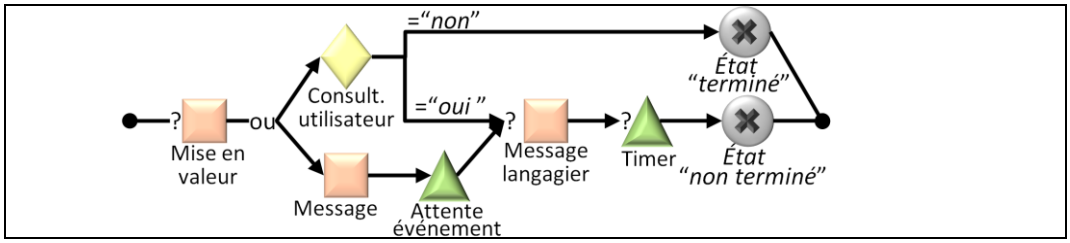

FIGURE 10 – Patron d'étape de pas à pas, en mode guidé.

5 Mise en œuvre du langage dans SEPIA

Nous avons mis en œuvre le langage aLDEAS ainsi que les patrons d'actions d'assistance qui le complètent dans l'environnement SEPIA (Specification and Execution of Personalized Intelligent Assistance).

5.1 L'éditeur d'assistance

L'éditeur d'assistance est un outil destiné aux concepteurs d'assistance. Il met en œuvre aLDEAS et permet de spécifier, pour une application-cible existante, un système d'assistance décrit par un ensemble de règles d'assistance respectant le patron de règles (cf. Figure 6). L'éditeur d'assistance propose une interface pour la création de chaque action élémentaire d'assistance présentée en section 3.3 (à l'exception de celles grisées sur les Figure 3 et Figure 4 qui ne sont pas mises en œuvre dans la version actuelle de SEPIA), ainsi que pour la création d'actions d'assistance qui instancient les patrons que nous avons proposés en section 4.

5.2 Les assistants épiphytes

Nous avons développé un ensemble d'assistants épiphytes, capables de réaliser dans une application-cible les actions élémentaires d'assistance proposées par aLDEAS et définies à

l'aide de notre éditeur. La Figure 11 présente les actions élémentaires d'assistance pouvant être réalisées par nos assistants épiphytes en fonction du type d'application-cible. Pour l'instant, nos assistants épiphytes sont capables d'agir principalement sur les applications Windows natives ou développées en Java, ainsi que sur les applications Web ouvertes avec les navigateurs Chrome ou Firefox.

		Action automatisée	Action mise en valeur	Action masquage	Action message	Action agent animé	Action ressources
Windows	Exécutables	✓	✓	✓	✓	✓	✓
	En Java	✓	✓	✓	✓	✓	✓
	Autres	✗	✗	✗	✓	✓	✓
Web	Firefox	✓	✓	✓	✓	✓	✓
	Chrome	✓	✓	✓	✓	✓	✓
	En Flash	✗	✗	✗	✓	✓	✓
	Autres	✗	✗	✗	✓	✓	✓

FIGURE 11 – Actions élémentaires d'assistance réalisables selon le type d'application-cible.

5.3 Le moteur générique d'assistance

Nous avons développé un moteur générique capable d'exécuter l'assistance spécifiée par le concepteur dans l'éditeur d'assistance. Pour réaliser les actions élémentaires, le moteur fait appel à l'un de nos assistants épiphytes. En ce qui concerne les actions qui instancient l'un des patrons d'actions composées que nous proposons, le moteur assure leur gestion et fait appel à un assistant épiphyte lorsque cela est nécessaire. La Figure 12 montre l'exemple des deux actions élémentaires, une mise en valeur et un message, déclenchées dans l'application-cible PhotoScape par la règle d'assistance R7 (cf. Figure 7 section 4.1).

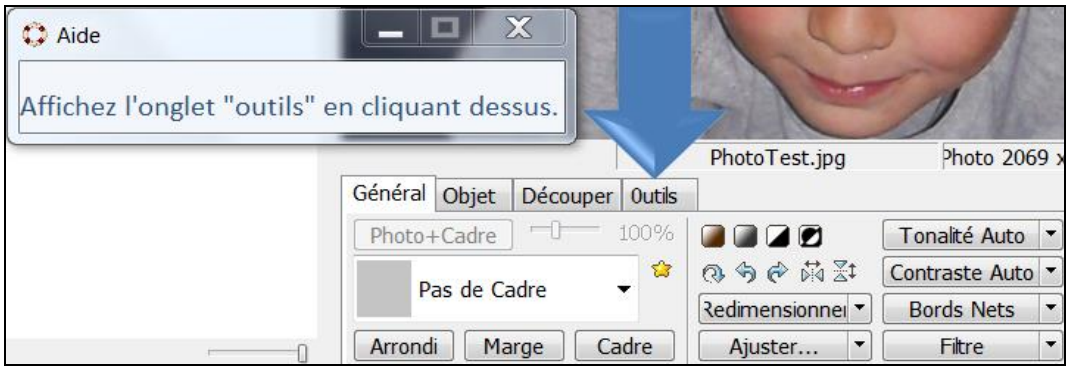

FIGURE 12 – Exemple de deux actions élémentaires d'assistance déclenchées dans PhotoScape.

6 Évaluation et discussion

Les propositions présentées dans cet article ont d'ores et déjà fait l'objet de plusieurs évaluations. En ce qui concerne la faisabilité de notre approche, elle est démontrée par la mise en œuvre que nous en avons faite à travers l'éditeur et le moteur générique d'assistance, complétés par nos collecteurs de traces et nos assistants épiphytes. Dans cette section, nous présentons les études menées pour évaluer la couverture du langage, ainsi que l'utilité et l'acceptation de l'assistance fournie aux utilisateurs finaux.

6.1 Couverture d'aLDEAS

Afin d'évaluer la couverture du langage que nous proposons, nous l'avons utilisé pour modéliser des systèmes d'assistance existants variés, représentatifs des types d'assistance fréquemment rencontrés. Ainsi, notre langage permet de modéliser les systèmes d'assistance souvent utilisés dans les applications Web contenant un formulaire : ils permettent par exemple d'expliquer à l'utilisateur où trouver les informations pour remplir un champ, ou le prévenir qu'un champ n'est pas rempli. Notre langage permet également de modéliser les systèmes d'assistance de type présentation guidée d'un logiciel ou d'une fonctionnalité. Par ailleurs, il existe de nombreux tutoriels proposés par des utilisateurs experts afin de guider un utilisateur pas à pas pour réaliser une tâche, à l'aide de copies d'écran annotées et de messages. Notre langage permet de modéliser de tels systèmes d'assistance avec l'avantage de le faire de façon plus intégrée à l'application-cible. Ainsi, les copies d'écran annotées pourront être remplacées par des actions de mise en valeur intégrée à l'application-cible.

Il existe néanmoins des systèmes d'assistance que notre langage ne permet pas de modéliser : c'est le cas des systèmes d'assistance très spécifiques à une application et requérant des informations non disponibles depuis l'extérieur de cette application. Par exemple, les moteurs de recommandations intégrés dans des applications de vente en ligne utilisent l'ensemble des informations sur les articles consultés ou achetés par tous les utilisateurs du site. Notre langage ne permet pas la consultation de l'ensemble de ces informations.

Nous avons également évalué le langage aLDEAS en le comparant aux approches existantes d'assistance spécifiée *a posteriori* pour une application-cible. En ce qui concerne l'approche proposée par (Richard, et al., 2004), aLDEAS permet également de modéliser de telles actions d'assistance, sous la forme d'une attente d'un événement de type clic sur un lien donné, consultation optionnelle des traces de l'utilisateur relatives à la navigation dans le site, puis action d'assistance de type message (contenant éventuellement des liens vers une autre page web ou ressource). aLDEAS propose également des actions élémentaires d'assistance de lancement de ressources sans passer par un lien. En ce qui concerne l'approche proposée par (Carlier, et al., 2010), aLDEAS permet la définition d'actions d'assistance impliquant des agents animés capables de se déplacer, de s'exprimer oralement et textuellement, ainsi que par des gestes et animations. Nous avons de plus facilité la définition de telles actions en proposant un patron d'action d'agent animé. Enfin, notre langage permet la création de règles de la forme <événement déclencheur, condition de déclenchement, action d'assistance, événement de fin> (cf. section 4.1) équivalentes aux règles utilisées dans les approches de (Paquette, 2012) et (Dufresne, et al., 2003), tout en permettant la définition de règles d'assistance plus complexes et variées. De plus, aLDEAS propose un large choix d'actions élémentaires d'assistance, ainsi que des patrons facilitant la définition d'actions d'assistance composées de nombreux éléments.

6.2 Utilité et acceptation de l'assistance proposée

Afin de tester l'utilité et l'acceptation de l'assistance que nous proposons, nous avons créé à l'aide de SEPIA un système d'assistance pour PhotoScape2. Ce système d'assistance permet de guider les utilisateurs dans la réalisation de chaque étape d'une tâche de correction des yeux rouges. Nous avons demandé à 200 personnes de corriger avec PhotoScape les yeux rouges sur une photo donnée, sans aide pour les 100 utilisateurs du groupe A, et avec l'aide spécifiée pour les 100 utilisateurs du groupe B. Pour le groupe A, on distingue 2 sous-groupes : A1 pour les 49 utilisateurs qui ont réussi à réaliser la tâche demandée sans assistance, et A2 pour les 51 utilisateurs ayant abandonné et à qui nous avons ensuite demandé de réaliser la même tâche avec l'assistance conçue. L'expérimentation était précédée et suivie d'un questionnaire. Les Figure 13 et Figure 14 présentent une partie des résultats de

[2] Vidéo de démonstration est disponible à http://liris.cnrs.fr/blandine.ginon/PhDWork.html

cette étude. On constate notamment que l'aide proposée permet de réaliser la tâche de manière deux fois plus rapide en moyenne. Dans le groupe A1, on note que la moitié des utilisateurs ayant réussi à réaliser la tâche sans assistance aurait pourtant souhaité recevoir de l'assistance. Beaucoup ont en effet précisé sur le questionnaire qu'ils auraient souhaité être guidés pour trouver plus rapidement les composants de PhotoScape permettant la correction des yeux rouges. Dans le groupe A2, les utilisateurs qui n'avaient pas réussi à réaliser la tâche demandée, on constate que 100% d'entre eux ont ensuite réussi grâce à l'assistance proposée et tous ont trouvé l'aide utile. Dans le groupe B, 97% des utilisateurs ont trouvé l'aide utile. L'assistance a été appréciée par les utilisateurs l'ayant testée, on note en effet que 92% des utilisateurs du groupe A2 et 87% des utilisateurs du groupe B l'ont appréciée. Ces résultats très satisfaisants nous conduisent à considérer que le système SEPIA permet de fournir aux utilisateurs finaux d'applications-cibles une assistance à la fois pertinente et efficace pour répondre aux besoins des utilisateurs, tout en étant bien acceptée par ces derniers.

	Effectif	Tâche sans assistance		Tâche avec assistance	
		Taux de réussite	Durée moyenne	Taux de réussite	Durée moyenne
Groupe A	100	49%	-	-	-
A1	49	100%	146,1 s	-	-
A2	51	0%	-	100%	72,15 s
Groupe B	100	-	-	100%	65,33 s

FIGURE 13 – Quelques résultats relatifs à l'expérimentation avec PhotoScape.

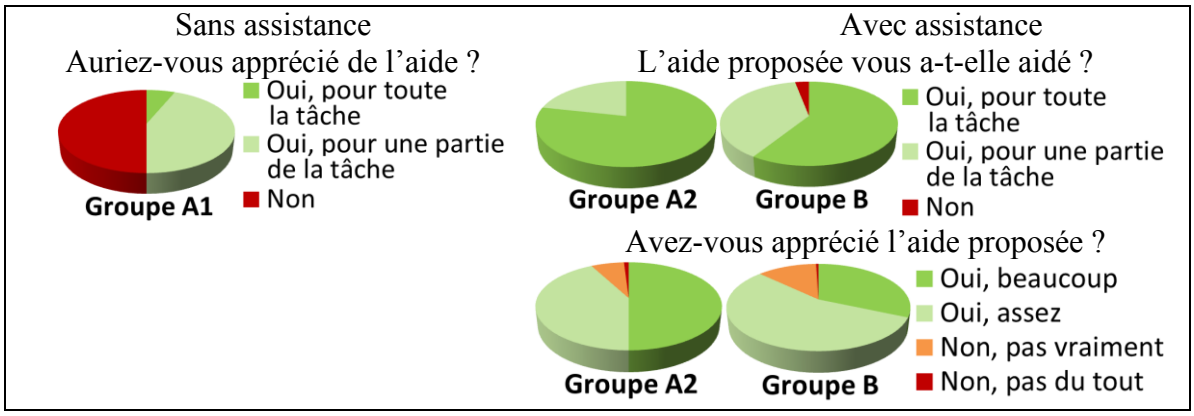

FIGURE 14 – Résultats de questions consécutives à l'expérimentation avec PhotoScape.

7 Conclusion et perspectives

Le langage aLDEAS et les patrons d'actions composées que nous avons présentés permettent la définition de systèmes d'assistance adaptés à leur application-cible. En adoptant une démarche entièrement épiphyte, nous permettons l'adjonction *a posteriori* de systèmes d'assistance à des applications existantes, non conçues spécifiquement pour l'intégration d'assistance, sans accès à son code source et sans connaissances particulières de programmation, par des concepteurs d'assistance qui ne sont pas nécessairement les concepteurs de l'application-cible et qui peuvent être des utilisateurs experts de l'application. Les outils qui mettent en œuvre aLDEAS permettent aux concepteurs d'assistance de spécifier des systèmes d'assistance capables de fournir aux utilisateurs finaux une assistance efficace

pour les aider à réaliser une tâche donnée, en particulier dans le cas de la découverte ou de l'utilisation occasionnelle d'une application-cible.

En exploitant les outils que nous avons développés, nous souhaitons désormais nous intéresser à la définition de systèmes d'assistance adaptés à des applications plus complexes, destinées à un public averti et requérant des connaissances spécifiques.

Par ailleurs, nous souhaitons également faciliter la tâche du concepteur d'assistance. En effet, nous souhaitons aider le concepteur à identifier les besoins d'assistance des utilisateurs finaux de l'application-cible. Pour cela, nous envisageons de proposer au concepteur des indicateurs calculés à partir de traces d'utilisation de l'application-cible. Nous souhaitons également aider le concepteur de l'assistance à améliorer le système d'assistance qu'il a conçu. Pour cela, nous souhaitons notamment permettre aux utilisateurs finaux de l'application-cible d'exprimer leur opinion vis-à-vis de l'assistance qui leur a été proposée. Ces retours d'utilisateurs, accompagnés d'indicateurs calculés à partir de traces d'utilisation de l'application-cible et du système d'assistance, seraient fournis au concepteur de l'assistance afin de lui permettre d'améliorer l'efficacité de son système d'assistance.

Références

DUFRESNE, A. (2001). Conception d'une interface adaptée aux activités de l'éducation à distance - ExploraGraph. In STE, p 301-319.

DUFRESNE, A., BASQUE, J., PAQUETTE, G., LÉONARD, M., LUNDGREN-CAYROL, K. & PROMTEP, S. (2003). Vers un modèle conceptuel générique de système d'assistance pour le téléapprentissage. In Sticef, p 57-88.

CARLIER, F. & RENAULT, V. (2010). Educational webportals augmented by mobile devices with iFrimousse architecture. In International Conference on Advanced Learning Technologies, Tunisia.

GAPENNE, O., LENAY, C. & BOULLIER, D. (2002). Defining categories of the human/technology coupling : theoretical and methodological issues. In ERCIM Workshop on User Interface for All, France, p 197-198.

GINON, B., CHAMPIN, P.-A. & JEAN-DAUBIAS, S. (2013a). Collecting fine-grained use traces in any application without modifying it. In Worshop EXPPORT of ICCBR, New-York, USA.

GINON, B., JEAN-DAUBIAS, S. & CHAMPIN, P.-A. (2013b). Mise en place d'un système d'assistance personnalisée dans une application existante. In IC, Lille, France.

PAQUETTE, G., PACHET, F., GIROUX, S. & GIRARD, J. (1996). EpiTalk, a generic tool for the development of advisor systems. p 349-370.

PAQUETTE, G. (2012). Référencement par compétence, recherche et assistance dans les environnements d'apprentissage et de travail. In TICE, Lyon, France, p 190-199.

RICHARD, B. & TCHOUNIKINE, P. (2004). Une approche centrée modèle pour la construction d'un système conseiller pour un site web. In IC, Lyon, France, p 151-162.