Des systèmes multi-agents épiphytes

Sylvain Giroux

IREMIA
Université de La Réunion
15 avenue René Cassin
97489 St-Denis
La Réunion, France
e-mail: giroux@iremia.fr

François Pachet

LAFORIA
Université Paris VI
4place Jussieu
75252 Paris Cedex 05,
France
e-mail: fdp@laforia.ibp.fr

Gilbert Paquette

LICEF
Télé-Université
1001, rue Sherbrooke est,
Montréal, Canada, H2X 3M4
e-mail: gilbert_paquette@
teluq.uquebec.ca

Mots-clés : réflexivité, introspection, multi-agents, systèmes conseillers, épiphyte

Résumé

Il existe une catégorie d'applications en informatique qui consistent non pas à créer un système d'information qui interagit directement avec un usager, mais plutôt à créer un système d'information qui observe le comportement d'un système d'information existant et qui réfléchit à partir de ses observations. Une analogie avec les plantes épiphytes est à la source de EpiTalk. EpiTalk est une plateforme qui permet de spécifier et de générer automatiquement des systèmes d'information capable de raisonner de manière introspective, distribuée et parallèle sur des architectures distribuées et parallèles. A cause de la nature même des applications envisagées, l'utilisation de systèmes multi-agents s'avérait incontournable. La structure d'un système d'information multi-agents épiphyte "se colle" au plus bas niveau sur la structure du système à observer. Aux niveaux supérieurs, la structure du système multi-agents épiphyte se calque sur la hiérarchie des tâches analysées. Par conséquent, la description de ces systèmes d'information est centrée sur un graphe des tâches. Un graphe des tâches exprime un point de vue particulier sur un système d'information hôte. Le lien entre le système hôte et le système épiphyte est réalisé par espionnage et assuré par la réflexivité organisationnelle. Finalement nous esquissons l'utilisation de Epitalk à travers deux exemples, la définition de déverminateurs configurables pour systèmes à acteurs et la définition de systèmes conseillers.

1. Introduction

Il existe une catégorie d'applications en informatique qui consistent non pas à créer un système d'information qui interagit directement avec un usager, mais plutôt à créer un système d'information qui observe le comportement d'un système d'information existant et qui réfléchit à partir de ses observations. En particulier, toute activité d'introspection appartient à cette catégorie. Par analogie avec la botanique, nous qualifions d'*épiphytes* de tels systèmes d'information: ceux-ci poussent sur les autres systèmes d'information.

Nous étudions actuellement les systèmes épiphytes dans le cadre des environnements de formation possiblement distribués [Pachet, et al., 1994a] [Paquette, et al., 1993] [Pachet, et al., 1993] et dans le cadre des environnements de programmation, en particulier le déverminage semi-automatique de systèmes mutli-agents parallèles et/ou distribuées [Giroux, et al., 1994ab]. Ces deux classes d'applications (environnements de programmation et environnements de formation) de par leurs caractéristiques s'avèrent des champs d'application tout indiqués pour les systèmes épiphytes. Dans le cas des systèmes de formation, un usager interagit avec un système et, à partir des interactions de l'usager et du comportement du système observé, un système conseiller épiphyte pilote l'usager sur la manière dont il utilise le système. Dans le cas du déverminage, le programme à déverminer devient le système à observer, alors que le déverminateur épiphyte observe et raisonne sur le comportement du programme à déverminer.

La plateforme EpiTalk [Pachet, et al., 1994b] permet de spécifier et générer des systèmes d'information

multi-agents épiphytes (SIMAE) pour des applications à base d'objets. Le cadre de travail proposé par EpiTalk emprunte à la réflexivité, aux systèmes multi-agents et aux systèmes à base de règles.
Tout d'abord, nous avons fixé comme contrainte que les applications à observer soient écrites indépendamment des SIMAE . Cela étant, il devient possible de greffer des SIMAE sur des systèmes déjà existants d'une part et d'autre part, de retirer les SIMAE lorsque leur travail d'introspection n'est plus requis, e.g. lorsque le programme est déverminé. Par conséquent, un SIMAE ne modifie, ni ne perturbe le comportement du système observé.

Ensuite, la nature même des SIMAE et des applications envisagées ont dicté l'utilisation comme cadre de travail de la réflexivité et des systèmes multi-agents. De son côté, la réflexivité s'avère un cadre de travail adéquat pour séparer clairement le système obervé du système observant, pour greffer le système observant sur l'observé et pour identifier les relations qui unissent l'observant et l'observé. La réflexivité permet de séparer clairement entre les opérations de différents niveaux. De l'autre côté, la distribution, le parallélisme, le non-déterminisme dans les opérations d'un usager et la multiplicité des niveaux d'expertise inhérents aux applications envisagées nous ont naturellement amené à privilégier une approche multi-agents. Ainsi un SIMAE est un système multi-agents greffé sur le système à observer par réflexivité.

Les systèmes multi-agents distribuent leur raisonnement à travers ses agents. Il devient alors parfois plus difficile de gérer et de comprendre le raisonnement d'un SIMAE. C'est pourquoi NeOpus [Pachet, 1992a] sert à formaliser et à organiser les connaissances à l'intérieur de EpiTalk. Pour cela, NeOpus y introduit les systèmes à base de règles et l'héritage entre base de règles. L'héritage entre base de règles [Pachet 1992b] permet de factoriser les règles. Une base de règles est construite comme sous-base d'une base de règles existantes. Dans notre contexte, ce mécanisme est utilisé pour raffiner ou spécialiser des bases de règles existantes et ainsi adapter le raisonnement à des situations particulières. Par exemple, les dévermineurs peuvent être configurés en fonction d'une application prise dans un contexte spécifique [Giroux, et al., 1994ab].

Finalement, les SIMAE sont générés automatiquement à partir d'une description du système à observer. Cette description est hiérarchique. D'un côté, une description hiérarchique permet de distinguer entre la localité et la globalité des actions entreprises par le système épiphyte. Ainsi le raisonnement n'est pas noyé dans un amas d'informations, les informations uniquement nécessaires sont clairement identifiées et enfin les agents d'un SIMAE peuvent rester physiquement proches des composantes observées dans les environnement physiquement distribués. De l'autre côté, une description hiérarchique permet aussi de distinguer au niveau du degré d'abstraction des informations requises pour le raisonnement. Par exemple, ce genre de distinction n'est pas possible dans le système développé par [Böcker et Herczeg, 1990]. Certes dans ce système, toute action de traçage peut être émise à partir du système traceur (le système observant) sans restriction sur la nature et la granularité de ces actions. Cependant les actions dans ce système sont toutes exprimées au même niveau, peu importe qu'elles concernent des détails d'implantations ou qu'elles concernent des états abstraits du programme que le traceur doit mettre en évidence.

Dans cet article, nous étayons d'abord notre analogie avec les plantes épiphytes. Nous décrivons ensuite EpiTalk. Finalement nous esquissons deux des applications en cours de développement. La première décrit un dévermineur taillé sur mesure pour une application multi-agents. La seconde décrit un système conseiller pour l'environnement de programmation de Smalltalk-80.

2. Une métaphore porteuse

Les plantes épiphytes sont des plantes qui croissent sur d'autres végétaux sans leur causer de préjudices, à l'opposé des plantes parasites. Aussi par analogie, nous qualifions les systèmes d'information multi-agents construits d'*épiphytes*. Cet adjectif, "épiphyte", résume bien les caractéristiques de la relation qui unit le système d'information observé -l'hôte- et un système d'information observant -le SIMAE-, soient

- le SIMAE ne peut exister sans hôte;
- l'hôte peut exister sans SIMAE;
- l'hôte et le SIMAE ont des existences indépendantes;

- le SIMAE ne porte pas préjudice à son hôte;
- le SIMAE constitue un système d'information assemblé à partir d'éléments distribués et autonomes, mais fortement reliés entre eux pour former un tout cohérent.

Enfin, les plantes épiphytes s'organisent souvent en véritables écosystèmes. Cette organisation en écosystèmes de SIMAE permet la coexistence de points de vue multiples sur une même situation, la structure de chaque SIMAE exprimant un point de vue particulier et partant, la communication d'informations entre SIMAE traduit les échanges entre points de vue différents sur une même problématique. Ainsi le terme "épiphyte" contribue à identifier une classe de systèmes.

3. Vocabulaire

Un SIMAE, est donc par métaphore botanique, un système d'information qui se greffe sans affecter son hôte. De plus, un SIMAE possède sa propre structure. Afin d'éviter les confusions de vocabulaire, nous utilisons dans cet article les termes suivants dans un sens relativement précis :

Agent : On appelle **agent** tout élément logiciel ou humain perçu comme faisant partie d'un système plus vaste. En particulier, les objets et les acteurs [Agha, 1986] sont perçus comme des agents.

Interaction : Une **interaction** est une transmission de message d'un agent à un autre.

Système ou **application :** Un système est vu comme un ensemble *cohérent* mais *non organisé* d'**agents**. Les agents sont cohérents car ils présentent à tout instant un état cohérent d'une situation quelconque. Il est non-organisé dans la mesure où il est non-déterministe. Aucune structure particulière n'existe pour l'activation des agents. Son évolution dépend étroitement des intra-actions et de ses interactions avec son environnement.

Hôte : L'hôte d'un SIMAE est le système d'information sur lequel est greffé le SIMAE.

Système d'information épiphyte : Un SIMAE est un système d'information qui raisonne et agit à partir de l'espionnage des interactions entre les agents qui composent son hôte.

Agent épiphyte : Un agent épiphyte est un agent évoluant dans un SIMAE.

Agent hôte : Un agent hôte est un objet appartenant à un hôte. En effet dans ce qui suit, un hôte est analysé sous forme d'organisation et ainsi les objets qui le composent sont perçus comme des agents.

4. EpiTalk

EpiTalk permet de spécifier et de générer des SIMAE pour toute application Smalltalk-80. EpiTalk est implémenté en Smalltalk-80. Il constitue un prolongement de Actalk pour sa partie acteurs [Briot, 1989], de ReActalk pour sa partie agents et réflexivité [Giroux, 1993] et de NéOpus [Pachet, 1992a] pour sa partie représentation des connaissances et raisonnement. Un SIMAE est greffé sur le système à observer par réflexivité. Les SIMAE sont générés automatiquement à partir d'une description du système à observer.

Dans cette section, nous proposons une architecture opérationnelle, comportant tous les maillons de la chaîne, de la conception abstraite du SIMAE à son implémentation. Cette architecture propose en effet à la fois un *formalisme de représentation* et un *vocabulaire* (basé sur les graphes hiérarchiques), ainsi qu'une *mécanique* d'interprétation de ces graphes. Ces aspects sont d'abord présentés à travers les propriétés et principes qui sous-tendent les SIMAE développés (§4.1). A l'instar de la plupart des systèmes de spécification, EpiTalk distingue entre la phase de spécification (§4.2) et la phase d'exécution (§4.3) pendant laquelle l'hôte est observé. Finalement, les tenants et aboutissants de l'espionnage sont décrits (§4.4).

4.1. Propriétés et principes des SIMAE

SIMAE et hôtes doivent posséder des propriétés d'indépendance les uns par rapport aux autres. Cette indépendance est à comprendre dans deux sens :

- au sens conceptuel :
  l'architecture du SIMAE est a priori indépendante de celle de l'hôte. Entre autres, leurs architectures ne sont aucunement semblables ou isomorphes. En particulier, la structure de l'hôte ne ressemble en rien à la structure des SIMAE greffés.

- au sens du génie logiciel :
  les hôtes ne doivent pas être conçus en fonction d'un SIMAE particulier. Cette contrainte est importante, car elle est le gage de la réutilisabilité des hôtes en tout ou en partie. Tout hôte programmé en fonction d'un SIMAE quelconque perd ses propriétés de réutilisabilité. Par exemple, construire un hôte en fonction d'un SIMAE déverminer présente peu de sens (§5.1). Inversement, un déverminer "en général" ne doit pas être conçu en fonction d'un système particulier à déverminer, bien qu'il puisse aussi être configuré sur mesure [Giroux, et al., 1994b].

Cette propriété d'indépendance amène donc à considérer hôte et SIMAE comme autonomes. Ceci constitue un premier argument en faveur des sytèmes multi-agents comme outil privilégié des SIMAE.

Maintenant si l'on s'intéresse aux environnements de formation, plusieurs facteurs conduisent tout aussi naturellement vers des architectures multi-agents:

- la multiplicité des niveaux d'expertise;
- la distribution physique de certains hôtes, e.g. les "classes virtuelles"[1] [Paquette, et al., 1993b];
- la distribution logique des hôtes, e.g. les systèmes multi-fenêtres;
- la nature même du processus humain de résolution de problèmes; même si les preuves sont présentées linéairement, leur découverte demande de nombreux allers-retours et s'éloigne nettement d'un processus linéaire.

Pour leur part, les environnements de programmation et de déverminage multi-agents manifestent également des caractéristiques de distribution physique, logique, de multiplicité d'expertise et de non-déterminisme et non-séquentialité du processus de résolution.

En outre, en délocalisant l'expertise, les systèmes multi-agents facilitent la tâche des concepteurs du SIMAE en proposant un cadre plus conforme à la réalité. Chaque agent épiphyte est chargé d'une tâche spécifique dont la granularité dépend de sa position dans la hiérarchie. Certains agents sont donc chargés d'une tâche bien délimitée (e.g. surveiller l'activité d'un seul agent du système hôte), d'autres voient leurs activités englober un ensemble de tâches (e.g. s'assurer que la démarche d'un usager corresponde à la démarche scientifique).

Toutes ces observations ont déterminé la structuration des agents épiphytes entre eux, ainsi que le traitement de l'information. Un SIMAE devient ainsi un organisme viable et cohérent à travers les quatre principes suivants :

(1) La description d'un SIMAE est tributaire d'un *graphe des tâches* hiérarchique. Un graphe des tâches "réifie" un point de vue donné sur l'activité de l'hôte. Il circonscrit si l'on peut dire le sujet d'étude du SIMAE. Ce sujet concerne tout aussi bien l'objet produit par le système à observer (e.g. la cohérence pédagogique des objectifs dans un logiciel de construction de curriculum [Paquette, et al., 1994]) que la cohérence de l'activité interne de l'hôte (e.g. le déverminage [Giroux, et al., 1994b]) ou la cohérence de son utilisation (e.g. est-ce que la démarche d'un usager respecte la démarche

[1] Contrairement aux classes traditionnelles où tous les étudiants sont regroupés physiquement dans un même lieu, les classes virtuelles regroupent les étudiants virtuellement, chacun participant à la vie scolaire à partir de sa station de travail située dans son domicile. Ce concept est particulièrement important dans les pays ou les régions à faible densité de population.

scientifique ? [Paquette, et al., 1993]).

(2) A l'exécution, les agents épiphytes sont organisés selon un graphe hiérarchique isomorphe au graphe des tâches.

(3) A l'exécution, les interactions entre les agents de l'hôte sont recueillies par des agents appelés "espions". Les espions s'insèrent dans l'hôte sans en troubler le fonctionnement et sans violer l'encapsulation de ses agents. Les espions réifient les interactions entre les agents et transmettent les messages espionnés aux agents épiphytes "terminaux". Le concepteur identifie les interactions particulières que chaque agent épiphyte terminal doit analyser.

(4) A l'exécution, les agents épiphytes se transmettent l'information de "bas en haut". Seuls les agents épiphytes terminaux (i.e. associés à des agents de l'hôte) reçoivent les interactions entre les agents de l'hôte via les espions. Chaque agent traite ces informations, puis les retransmet transformées et analysées au niveau supérieur, et ainsi de suite.

Ces quatre principes soutiennent toute l'architecture qui s'y conforme aveuglément.

4.2. La spécification des SIMAE

EpiTalk repose sur la manipulation de trois graphes qu'il est important de bien distinguer afin de comprendre la suite de la description. Il s'agit de la réification de l'hôte sous forme d'organisation, du graphe des tâches qui décrit abstraitement les SIMAE et du graphe des agents épiphytes qui représente un SIMAE pendant son activation. Dans cette section, nous détaillons la spécification et le rôle de chacun.

4.2.1. La réification de l'hôte

Un hôte est formé d'un groupe d'objets/agents. Ce groupe est réifié par un graphe. Ce graphe n'existe qu'à l'exécution et évolue avec l'hôte. Les nœuds correspondent aux agents hôtes existants. Chaque fois qu'un acteur est créé, un nœud correspondant est ajouté. Les arcs représentent les réseaux de communication. Au niveau de l'implantation, il suffit de connaître les objets faisant partie de l'hôte; les réseaux de communication peuvent être déduits à l'aide des messages espionnés et de facilités réflexives [Foote et Johnson, 1989] de Smalltalk-80. Pour l'instant, les liens entre les noeuds n'interviennent pas dans la construction et le raisonnement d'un SIMAE.

4.2.2. Le graphe des tâches

La description d'un SIMAE est centrée sur un *graphe des tâches*. Ce graphe sert à spécifier l'architecture du SIMAE de manière abstraite. Tout SIMAE est généré à partir de ce graphe:

le graphe des agents épiphytes est isomorphe au graphe des tâches.

Plus précisément, cet isomorphisme a deux conséquences :

- A chaque tâche du graphe des tâches est associé, à l'exécution, un agent épiphyte et un seul.

- La structure du graphe des agents épiphytes est la même que celle du graphe des tâches, i.e. les agents épiphytes auront la même structure hiérarchique que les tâches qu'ils surveillent.

D'une part, le graphe des tâches explicite le point de vue qu'a le concepteur de l'hôte et, d'autre part, ce point de vue détermine le processus d'analyse, réalisé par les agents épiphytes, des interactions se produisant à l'intérieur de l'hôte.

Un graphe des tâches possède plusieurs caractéristiques :

- Le graphe est hiérarchique. Il n'y a pas de cycles.

- Les "racines" du graphe sont les nœuds sans supérieur hiérarchique.
- La relation père-fils est une relation de hiérarchie de parties : un nœud non terminal "contient" tous ses fils, et les descendants de ses fils, récursivement.

Nous distinguons trois sortes de nœuds dans ce graphe :

1- les nœuds "terminaux", ou les feuilles.

   Ces nœuds sans fils possèdent un statut particulier car les agents de l'hôte sont associés à ces nœuds. Seuls les nœuds terminaux du graphe des tâches se voient associer un ou plusieurs agents hôtes. Par convention, on représente les nœuds terminaux entre crochets, [].

2- Les nœuds non terminaux. Ce sont les nœuds qui ne sont pas des feuilles. On distingue encore deux sous-cas :
   2a- les nœuds non terminaux simples
   2b- les nœuds "étoile". Les nœuds "étoile" représentent des hiérarchies de partie dont le nombre de composants (de 0 à n) est inconnu a priori. A chaque création d'un agent hôte concernant la tâche, le sous-arbre correspondant est ré-instancié. Par convention, ces nœuds sont suffixés d'une étoile "*" .

Le terme "graphe des tâches" est volontairement général pour inclure toutes les descriptions qu'un programmeur désire énoncer sur un hôte existant. Cette description peut être une description des *tâches* à effectuer par un utilisateur ou par l'hôte[2] ou bien une description hiérarchique des *résultats* à produire. La sémantique d'un graphe des tâches est donc entièrement déterminée par le concepteur.

La mécanique d'instanciation des agents épiphytes est intimement liée au graphe des tâches et à sa structure. En effet, ce graphe guide la génération d'un SIMAE au moment du lancement d'un hôte et au fur et à mesure de son évolution. Plus précisément, le graphe des tâches remplit deux fonctions :

- organiser à l'exécution les agents épiphytes dans une hiérarchie;

- pour chaque agent épiphyte, spécifier les agents hôtes à surveiller et, pour chaque agent hôte, spécifier les messages à capter.

Ainsi EpiTalk n'exploite actuellement dans un graphe des tâches qu'un seul type de lien, celui de hiérarchie de parties, ou d'aggrégation.

Bien sûr, il peut exister d'autres types de liens dans un graphe des tâches, comme les liens de précédence temporelle (telle tâche doit être effectuée avant telle autre), ou des liens conditionnels. Ces liens se traduisent alors dans la spécificité des opérations effectuées par les agents épiphytes. Pour l'instant, une contrainte de précédence entre deux tâches `t1` et `t2` s'exprime dans la tâche qui contient à la fois `t1` et `t2`. Le comportement de l'agent épiphyte associé à cette tâche spécifie leur ordonnancement correct ainsi que les actions à prendre en cas de non-respect. Les liens du graphe des tâches ne sont pas exploités actuellement dans le raisonnement des agents épiphytes.

Bien évidemment, il existe autant de graphes des tâches pour un même hôte qu'il existe de points de vue sur cet hôte. Comme leur architecture multi-agents permet aux SIMAE d'opérer en parallèle, il est donc possible d'analyser en parallèle un hôte sous plusieurs points de vue.

4.2.3. Le graphe des agents épiphytes

Le dernier graphe est le graphe des agents épiphytes. Il n'existe de toute évidence qu'à l'exécution. Les agents de ce graphe sont générés automatiquement par EpiTalk à partir du graphe des tâches et en fonction

[2] Par exemple, les tâches au sens de Chandrasekaran [Chandrasekaran 87].

des agents de l'hôte. Conformément à l'hypothèse fondamentale de EpiTalk, la structure du graphe des agents épiphytes est isomorphe à celle du graphe des tâches. Cependant les graphes n'en demeurent pas moins tout à fait différents et distincts, le premier entrant en jeu pendant l'exécution, le second, pendant la phase de description.

Cet isomorphisme est réalisé simplement : chaque tâche (nœud du graphe des tâches) est associée à un (et un seul) agent épiphyte. Si le nœud est terminal, l'agent épiphyte est terminal. Si le nœud est non terminal, l'agent épiphyte correspondant est non terminal. De plus, les liens de hiérarchie du graphe des tâches sont transposés dans le graphe des agents épiphytes.

4.3. Génération d'un SIMAE

Comme nous l'avons vu, l'architecture d'un SIMAE est entièrement spécifiée par le graphe des tâches. Au démarrage de l'hôte, le SIMAE est "instancié" à partir du graphe des tâches et sa structure est isomorphe à celle du graphe des tâches. Cependant à cause des tâches étoiles, un SIMAE n'est pas toujours entièrement généré. En effet, pour ces tâches, l'instanciation "paresseuse" d'un niveau supplémentaire (le fameux "0 à n") dans le SIMAE est déclenchée par la création dans un contexte donné de certains agents hôtes. Le contexte de création est en effet tout aussi important que le nouvel agent hôte. Connaître les seules classes et méthodes entraînant la création de nouveaux agents hôtes ne suffit guère, puisque le contexte de création peut s'avérer primordial pour l'analyse. Par exemple, le contexte de la création des fouineurs en Smalltalk-80 peut donner de sérieuses indications sur les intentions d'un usager. Les tâches étoiles couplés à la détection en contexte des nouveaux agents hôtes permettent d'exprimer de telles subtilités.

La mécanique d'instanciation du graphe des agents épiphytes est basée sur un principe unique: chaque création d'un nouvel agent hôte provoque le parcours du graphe des agents épiphytes existants. Ce parcours a pour effet de connecter les agents épiphytes existant au nouvel agent hôte si nécessaire, et, le cas échéant, d'instancier les agents épiphytes étoile. Plus précisément, à chaque instanciation d'un nouvel acteur, EpiTalk effectue les deux tâches :

- il connecte les agents épiphytes existants intéressés par l'acteur;
- il instancie les tâches étoiles qui doivent l'être.

Ces deux opérations sont effectuées en même temps de la manière suivante :

> Le graphe des agents épiphytes est parcouru (en profondeur d'abord) à partir du père de l'agent épiphyte qui a détecté le nouvel agent hôte. Si l'agent hôte a été créé à partir d'un bouton dans l'éditeur général, l'agent épiphyte correspondant est tout simplement l'agent épiphyte racine. Le graphe des agents épiphytes est donc entièrement parcouru. Au contraire, si la création de l'agent hôte est repérée par un agent épiphyte (terminal) déjà existant, alors uniquement le sous-graphe des tâches de cet agent épiphyte est parcouru et mis-à-jour.

Pour chacun des agents épiphytes rencontrés lors de ce parcours les opérations suivantes sont effectuées :

(i) Si la tâche de l'agent épiphyte est une tâche terminale, et que l'agent épiphyte est intéressé par ce nouvel agent hôte, alors l'espion dédié à cet agent hôte est connecté à l'agent épiphyte. Cette connexion aura pour effet d'informer l'agent épiphyte des envois des messages identifiés dans la tâche terminale comme pertinents par le concepteur.

(ii) Si la tâche de l'agent épiphyte est une tâche non terminale simple, alors le parcours est simplement poursuivi dans les sous-agents épiphytes.

(iii) Enfin, si la tâche de l'agent épiphyte est une tâche étoile et que cette tâche est concernée par le nouvel agent hôte, alors la tâche est ré-instanciée totalement. et seul le nouveau sous-graphe des agents épiphytes est ensuite parcouru de manière récursive. Une tâche étoile est "concernée" par un

nouvel agent hôte si au moins une de ses sous-tâches l'est (c'est donc un calcul récursif qui s'arrête aux tâches terminales).

La détection dynamique des agents hôtes posent certains problèmes d'implémentation. En effet, la représentation réflexive d'un groupe doit correspondre exactement à la structure du groupe décrit. Aussi ces créations doivent-elles être repérées afin de pouvoir greffer les espions et lier les acteurs aux agents épiphytes. Ces questions sont discutées dans [Pachet, et al., 1994b].

4.4. L'espionnage des interactions: description et mise en oeuvre

Dans les systèmes à objets, les interactions correspondent aux transmissions de message. De par l'encapsulation des agents hôtes, les messages sont la seule manière dont les agents peuvent interagir entre eux. Par conséquent, l'espionnage des interactions se traduit au niveau de l'implémentation par l'espionnage des transmissions de message.

Trois questions se posent alors:

1- Quelles sont les limites d'un système d'agents ? Quels acteurs en font partie ? Faut-il inclure les acteurs humains qui interagissent à l'aide d'une interface ?

   En pratique, seules les interactions entre un usager et un système d'information qui se traduisent par un envoi de message à un agent hôte peuvent être recueillies par un système d'information. En effet, du point de vue du système d'information, seuls les *messages transmis aux agents* après une action de l'utilisateur (et non pas directement les actions de l'utilisateur) sont accessibles. Ainsi ne pourront être observés les clics ou les actions de l'utilisateur qui ne sont pas interprétés par le système d'information hôte en termes de transmissions de message. Par conséquent dans EpiTalk, la spécification d'un espion comporte, d'une part, les sélecteurs des messages dignes d'intérêt (pour entre autres minimiser le flot de communication entre un hôte et un SIMAE) et, d'autre part, les agents épiphytes à qui seront acheminés les messages observés.

2- Quels sont les éléments pertinents d'une interaction ?

   Pour l'instant dans EpiTalk, la réification d'une interaction comprend l'émetteur, le destinataire, le moment, ainsi que le message. Eventuellement, les interactions pourront être exprimées en termes d'actes de langages [Searle, 1969].

3- A quel moment, les interactions sont-elles captées ?

   Les messages peuvent être observés à leur émission, pendant leur transit ou à leur réception. Actuellement EpiTalk permet de les observer à leur réception. Peu importe le moment, leur représentation et leur analyse devront résoudre les trois problèmes que sont l'ordre d'activation, le couplage des transactions et l'ordre d'arrivée des messages [Manning, 1987].

5. Applications

Epitalk est utilisé dans le cadre de deux champs d'application distincts. D'un côté, EpiTalk est utilisé pour générer des environnement de formation. Voici quelques applications en cours de développement

- un système de spécification de cours: le sujet étudié par le SIMAE est le résultat produit par le système, soit la cohérence du cursus [Paquette, et al., 1994];

- un environnement d'apprentissage visant la découverte de la loi des gaz parfaits: le sujet étudié est la navigation de l'étudiant à travers cet environnement d'apprentissage [Paquette, et al., 1993];

- un système tuteur multi-agents appliqué à la géométrie.

- le développement de “classes virtuelles”.

De l’autre côté, Epitalk est utilisé pour générer des environnements de déverminage semi-automatique multi-agents pour systèmes à acteurs et à agents [Giroux, et al., 1994ab]. Bien évidemment, développer deux applications des systèmes multi-agents épiphytes en parallèle facilmite l'identification et l’abstraction des caractéristiques et de la généricité de tels systèmes. Dans cette section, nous esquissons un exemple pour chaque champ d'application.

5.1. Des dévermineurs épiphytes configurables

Traditionnellement la trace d'un programme est réalisée en ouvrant un flot et en ajoutant des descriptions des événements dans ce flot au fur et à mesure qu'il se produisent. Cette approche était acceptable à cause de la nature séquentielle des programmes: il existait un ordre total sur les événements et cet ordre correspondait bien à l'ordre causal des événements. Cependant le parallélisme dans les systèmes acteurs rend cette approche inadéquate. Dans une architecture distribuée non seulement l'ordre des messages est non-déterministe, mais aussi l'ordre d'arrivée de la description des événements au flot de la trace. C'est pourquoi nous avons cherché une manière distribuée de représenter et d'analyser ces événements. De plus, toujours à cause de ce même non-déterminisme, une trace doit être un objet plus structuré qu'un simple flot. Cette trace doit refléter l'ordre causal des événements. Enfin, les outils de déverminage se bornent en général à rassembler et à organiser une trace des interactions et opérations réalisées à l'intérieur des systèmes à déverminer. Nous développons actuellement de dévermineurs épiphytes capables non seulement de recueillir une trace, mais surtout configurables sur mesure pour le déverminage d'un hôte. En voici un exemple appliqué au crible d'Erathostène.

Le crible d'Erathostène [Abelson et Sussman, 1985] sert à générer les nombres premiers de 2 à N. Tout crible est associé à un nombre premier. Ces cribles sont chaîné selon ordre croissant de leur nombre premier pour former un pipeline. Le flot des entiers de 2 à N sert d'entrée au premier crible. Chaque crible retire de ce flot les multiples de son nombre premier. Si un entier atteint le bout du pipeline, il est premier et un crible correspondant est créé et chaîné avec le pipeline des cribles.

Un graphe des tâches possibles de cette application est donné en Figure 1. Le système à acteurs y est étudié du point de vue du bon enchaînement des opérations. Par exemple, la tâche étoile Les cribles * vérifie le bon chaînage des cribles dans le pipeline, alors que la tâche [Criblage] s'assure que les entiers sont reçus en ordre croissant, ce qui est essentiel pour le bon fonctionnement du crible dans son ensemble. Les traitement locaux restent donc locaux et les traitements globaux sont exprimés au bon niveau. La Figure 2 montre comment le SIMAE a été généré par EPiTalk et a suivi correctement l'évolution de son hôte en concordance avec les tâches étoiles.

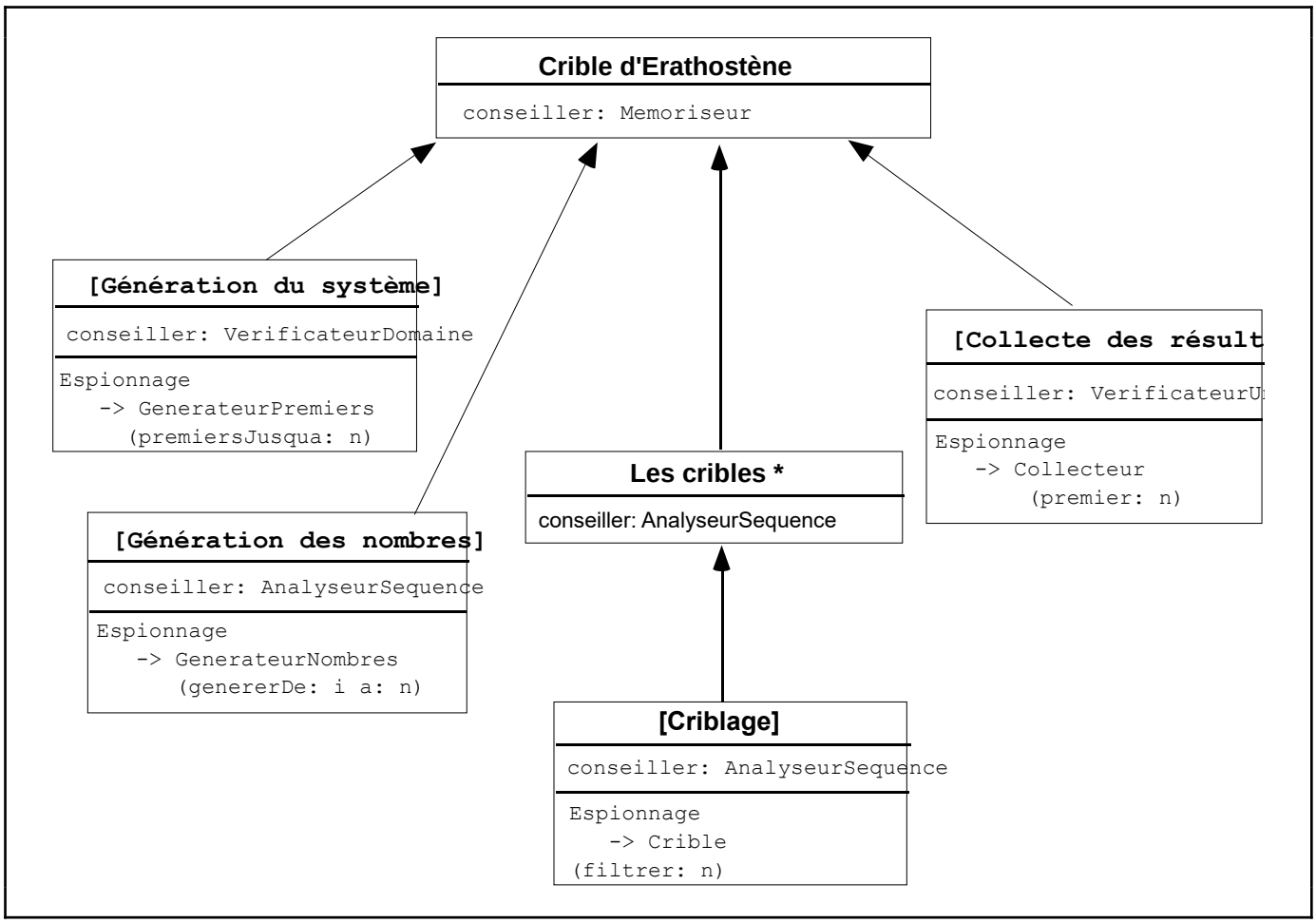

Figure 1. Un graphe des tâches pouvant être associé au crible d'Erathostène.

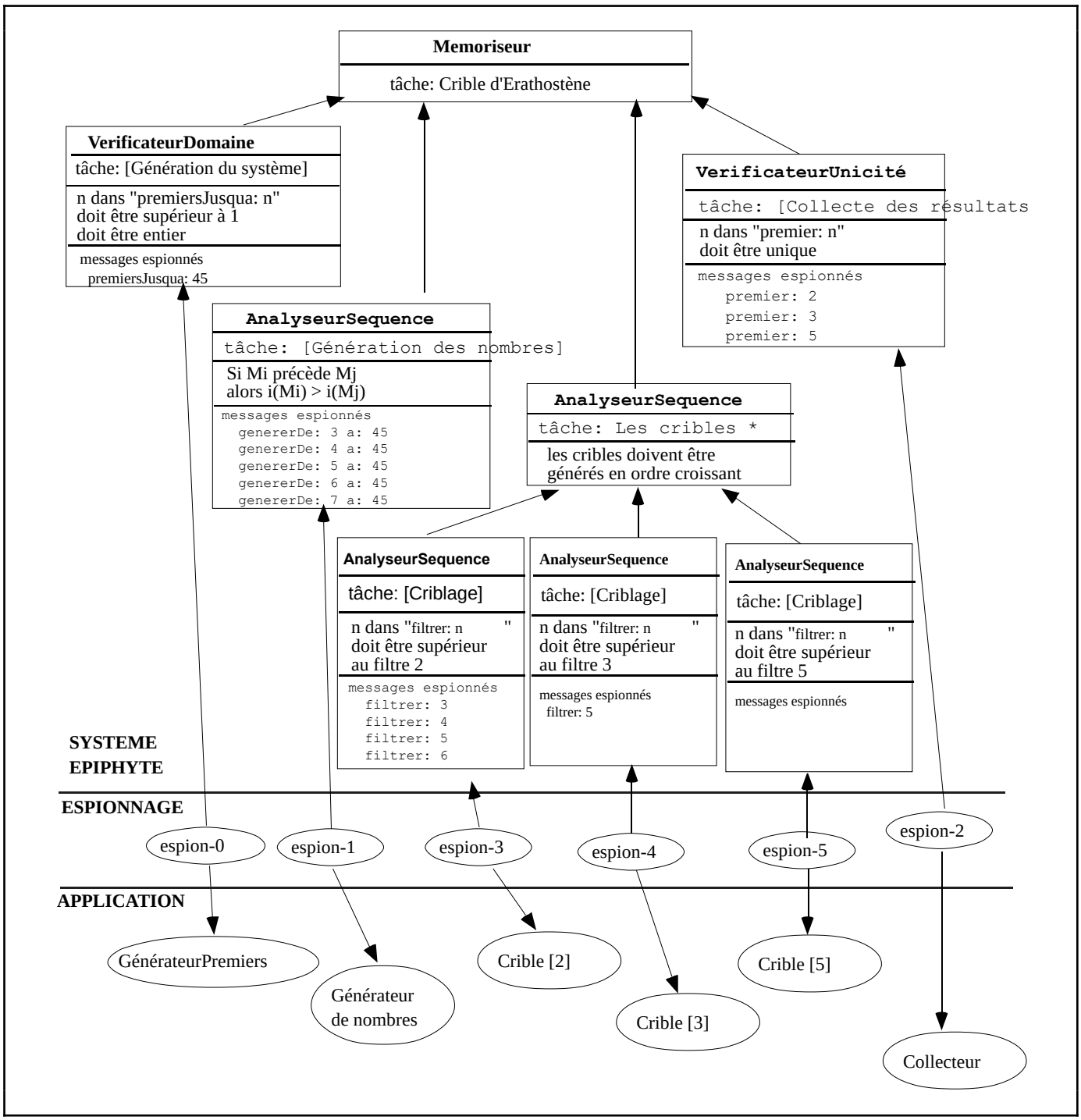

Figure 2. Le SIMAE et les connaissances de chaque agent épiphyte.

5.2. Des systèmes conseillers épiphytes

EpiTalk est aussi actuellement utilisé pour définir des systèmes conseillers pour l'environnement de programmation de Smalltalk-80. Il s'agit d'un exemple d'applications d'EpiTalk sur un système hôte déjà existant. Le graphe des tâches présente Smalltalk-80 en tant qu'environnement de programmation en général (Fig. 3). Le branchement de ce SIMAE conseiller dans l'environnement de programmation de Smalltalk-80 a nécessité très peu de modifications des méthodes existantes[3]. Des SIMAE conseillers dédiés à un problème particulier à résoudre en Smalltalk-80, e.g. la résolution de problèmes de géometrie, peuvent aussi cohabiter avec ce conseiller général.

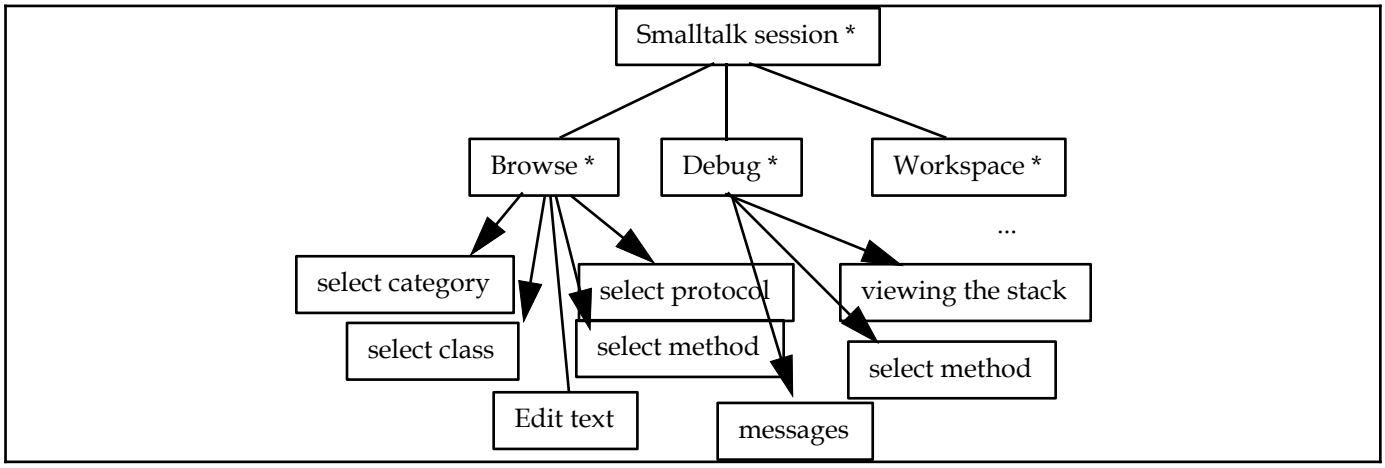

Figure 3. Un point de vue particulier sur la programmation en Smalltalk-80 en tant qu'activité.

6. Conclusion

Pour raisonner de manière introspective, distribuée et parallèle sur des architectures distribuées et parallèles, nous avons défini et développé EpiTalk. A cause de la nature même des applications envisagées, l'utilisation de systèmes multi-agents s'avérait incontournable. Les SIMAE reposent sur une anologie avec les plantes épiphytes. La structure d'un SIMAE "se colle" sur la structure de son hôte au niveau inférieur. Aux niveaux supérieurs, la structure du SIMAE se calque sur les tâches analysées. EpiTalk est une plateforme qui permet de spécifier et de générer automatiquement de tels systèmes d'information. La description des SIMAE est centrée sur un graphe des tâches. Un graphe des tâches exprime un point de vue particulier sur un système d'information hôte. Le lien entre le système hôte et un SIMAE est réalisé par espionnage et assuré par la réflexivité organisationnelle. Finalement l'utilisation de Epitalk est esquissée à travers deux exemples, la définition de dévermineurs configurables pour systèmes à acteurs et la définition d'un système conseiller pour l'environnement de programmation de Smalltalk-80.

Bibliographie

[Abelson et Sussman, 1985] Abelson, Harold; and Gerald Jay Sussman, *Structure and Interpretation of Computer Programs*, Cambridge, Mass., The MIT Press, 1985, 542 p.

[Agha, 1986] Agha, Gul, *Actors: A Model of Concurrent Computation in Distributed Systems*, The MIT Press, 1986.

[Böcker et Herczeg 90]. Böcker H.-D, Herczeg J. *What tracers are made of*, OOPSLA/ECOOP '90 Proceedings, October 1990, Ottawa, Canada, Sigplan Notices, vol. 25, no 10, pp. 89-99.

[Briot, 1989] Briot, Jean-Pierre, *Actalk: a Testbed for Classifying and Designing Actor Languages int the Smalltalk-80 Environment*, ECOOP '89 Proceedings, Nottingham, Angleterre, 10-14 juillet 1989, pp. 109-129.

[3] Jusqu'à présent, les seules modifications nécessaires visaient à modifier les méthodes de création des outils de programmation pour qu'elle rende le nouvel outil et non plus `self`, la valeur par défaut.

[Chandrasekaran 87] Chandrasekaran B. *Towards a functional architecture for intelligence based on generic information processing tasks*. Proc. of the Tenth IJCAI, Milan (Italy), Vol. 2, 1987, pp. 1183-1192.

[Foote et Johnson, 1989] Foote, Brian et Ralph E. Johnson, *Reflective Facilities in Smalltalk-80*, OOPSLA Proceedings, 1-6 oct. 1989, New Orleans, Louisiana, SIGPLAN Notices, vol. 24, no 10, oct. 1989, pp. 327-335.

[Giroux, 1993] Giroux, Sylvain, *Agents et systèmes, une nécessaire unité*, thèse de Ph. D. en informatique, Département de mathématiques et de recherche opérationnelle, Université de Montréal, août 1993, 246 p.

[Giroux, et al., 1994a] Giroux, Sylvain, François Pachet Pachet and Jocelyn Desbiens, *Debugging Multi-Agent Systems: a Distributed Approach to Events Collection and Analysis*, soumis à CWDAI'94, Canadian Workshop on Distributed Artificial Intelligence, Banff, Alberta, Canada, 16 mai 1994.

[Giroux, et al., 1994b] Giroux, Sylvain, François Pachet et Jocelyn Desbiens, *Customized Distributed Debuggers*, soumis à TOOLS USA '94, International Conference and Exhibition on Technology of Object-Oriented Languages and Systems Proceedings, 1994, Santa Barbara, California, August 1-5, 1994.

[Manning, 1987] Manning, Carl R., *Traveler: The Apiary Observatory*, ECOOP Proceedings, 1987, Paris, 15-17 juin, pp. 97-105.

[Pachet, 1992a] Pachet, François, *Représentation de connaissances par objets et règles: le systèmeNéOpus*, thèse de doctorat, LAFORIA, Université Pierre et Marie Curie, France, septembre 1992.

[Pachet 1992b] Pachet, F. Rule Base Inheritance. Conference "Object-based representations", La Grande Motte, France, June 1992.

[Pachet, et al., 1993] Pachet, François, Sylvain Giroux et Gilbert Paquette, *EpiTalk, une architecture et une implantation d'un système conseiller générique*, Rapport de recherche, LICEF, Télé-université, Montréal, nov. 1993.

[Pachet, et al., 1994a] Pachet, François, Sylvain Giroux and Gilbert Paquette, *Pluggable Advisors as Epiphyte Systems*, soumis à CALISCE '94, International Conference on Computer Aided Learning and Instruction in Science and Engineering Proceedings, 31 août, 1-2 sept. 1994, Paris, France.

[Pachet, et al., 1994b] Pachet, François, Sylvain Giroux and Gilbert Paquette, *Epiphyte Programming in Smaltalk-80*, soumis à TOOLS USA '94, International Conference and Exhibition on Technology of Object-Oriented Languages and Systems Proceedings, 1994, Santa Barbara, California, August 1-5, 1994.

[Paquette, et al., 1993] Paquette, Gilbert, Françoise Crevier, Sylvain Giroux et François Pachet, *Méthodes et outils de développement de systèmes conseillers dans les environnements de formation*, Rapport de recherche du LICEF, Télé-Université, Montréal, décembre 1993.

[Paquette, et al., 1993b] Paquette, Gilbert, Gilles Bergeron and Jacqueline Bourdeau, *The Virtual Classroom revisited.*, Conference on TeleTeaching Proceedings, 1993, Trondheim, Norway, août 1993.

[Paquette, et al., 1994] Paquette, Gilbert, et al., *Design of a Knowledge-based Didactic and Generic Workbench*, soumis à CALISCE '94, International Conference on Computer Aided Learning and Instruction in Science and Engineering Proceedings, 1994, August 31, September 1-2 1994, Paris, France, pp. submitted to.

[Searle, 1969] Searle, J., *Speech Acts: an essay in the philosophy of language*, Cambridge University Press, 1969.