Dominique CARADANT

Laboratoire d'Informatique
Appliquée à l'Architecture
Ecole d'Architecture de Toulouse
Laboratoire des Systèmes
Informatiques
Université Paul Sabatier

L'approche objet dans la CAO en architecture état de l'art et propositions

Résumé

Cette communication expose plus particulièrement les notions de representation de connaissances dans les systèmes de C.A.O en Architecture.Il traite plus particulièrement des techniques à base de structures de "frames" et de langages orientes objets. La nécessité d'avoir une structure dynamique pouvant nous aider à représenter la sémantique des "objets" manipulés dans la phase de conception et la volonté de disposer d'un interface utilisateur convivial nous ont conduit vers un nouveau type de langages. Ils allient la puissance declarative des "frames" a la modularité des structures de contrôle des langages orientés objets.

Après avoir fait un état de l'Art de l'introduction de l'intelligence artificielle dans la CAO en Architecture, le lecteur se familiarisera avec l'approche objet et la programmation par envois de messages. Nous montrerons aussi les nouvelles possibilités offertes par l'intégration de ce type de programmation à l'interieur d'une structure de donnees de type "FRAMES". Finalement, nous décrirons les dernieres phases d'évolution du langage LORCA et ses extensions actuelles.

Abstract

This communication expose the representing notions of knowledges in the C.A.D. systems, and particularly in Architecture. It deals with technics based on structures with "frames" and with Objects Oriented languages . The need of having a dynamic structure able to help us on representing the semantic of the "objects" manipulated in the designing phase, and the purpose of having at our disposal a convivial using interface lead us to a new type of languages. They ally the declarative power of the "frames" with the modularity of the controlling structures in the Objects Oriented languages.

After making a state of the Art in the introduction of the artificial intelligence in the C.A.D., in Architecture, the reader will master the approach object and the programming by dispatching messages. We will show too the new possibilities given by the integration of this programming inside a structure with data of type "Frames". Finally, we will describe the last developping stages of the language LORCA and its present extensions.

1 LA CONCEPTION ET DE LA C.A.O EN ARCHITECTURE

1.1 La C.A.O en architecture

Nous adopterons comme définition du terme CAO : "ensemble des techniques informatiques qui permettent l'exploitation, la mémorisation et la réexploitation du savoir-faire du concepteur". C'est dans cette optique et avec la volonté de fournir des outils capables de répondre à ces objectifs que nous avons réalisé cette étude.

Lorsque l'objet à concevoir est complexe, et ce sera toujours le cas lorsque l'on utilisera la CAO, le concepteur ne pourra en assurer la synthèse globale. Il va décomposer ce produit en sous-ensembles qui seront assemblés par la suite. Cette approche conceptuelle fait appel à la notion de décomposition hiérarchique d'un produit, c'est-à-dire à la structuration arborescente des éléments constitutifs du projet final. Cette approche engendre deux démarches de résolution : la démarche ascendante et la démarche descendante

La démarche descendante ou analytique consiste à définir des spécifications, niveau par niveau, en partant de l'objet et en descendant vers les constituants élémentaires.

La démarche ascendante consiste en l'élaboration à partir des constituants physiques élémentaires de blocs plus importants pour aboutir "in fine" aux spécifications souhaitées.

L'évolution rapide des systèmes informatiques, ces vingt dernières années, nous avait permis d'espérer que l'ordinateur allait entrer de manière significative dans le monde de la conception architecturale.

L'analyse des systèmes de CAO mis à la disposition des architectes montre qu'ils traitent pour la plupart uniquement de DAO ; ceux qui s'attribuent l'appellation de "Systèmes de CAO" ne s'intéressent, d'ailleurs, avec plus ou moins de réussite, qu'au Rendu Assisté par Ordinateur, production de plans, de pièces écrites , de perspectives lignes cachées en fausses couleurs... L'introduction de la CAO en architecture, permettant la maîtrise de projets de plus en plus complexes, aurait pu avoir un impact équivalent à celui réalisé dans la conception en électronique.

Sans nier l'intérêt que peut susciter l'introduction de nouvelles techniques de représentation graphique dans la phase de "rendu" d'un projet architectural, nous nous intéresserons dans cette étude à l'utilisation de l'ordinateur, à l'ensemble des phases du "projet architectural" et plus particulièrement à la phase de conception . il s'agit pour nous de définir les moyens à mettre en oeuvre pour faire de l'ordinateur un " assistant d'aide à la conception intelligent".

1.2 L'Intelligence Artificielle et la C.A.A.O

Les recherches actuelles en Intelligence Artificielle appliquée à la CAO en architecture s'orientent sur les axes suivants :

- Les systèmes de type résolution de problèmes dans le domaine de l'architecture
- Les systèmes experts généraux
- les sytemes à raffinement automatique bases sur un système à base de frames

1.2.1 Les systemes de type résolution de problèmes dans le domaine de l'architecture

Les premiers systèmes d'intelligence artificielle se sont focalisés sur les techniques de résolution de problèmes. il est donc naturel qu'un certain nombre de propositions aient été faites dans cette direction <KALAY 85>.

Ces propositions tentent de faire le lien entre les techniques de résolution de problèmes telles qu'elles sont abordées dans les logiciels d' I.A : techniques d'exploration , heuristiques ou non , d'un graphe d'états [1]. Le cas de la conception en architecture est beaucoup plus complexe : si la conception fait appel aux techniques de résolution de

[1] Comme cela est fait dans la plupart des jeux et les premiers systèmes généraux de résolution de problèmes. L'état courant étant, par exemple, la position des pièces sur l'échiquier et les transitions générant d'autres états les coups possibles

problèmes, il est difficile d'assimiler le "domaine de résolution" du concepteur à celui d'un problème de mathématiques ou d'un jeu d'échec.

Des spécificités fondamentales empêchent cette assimilation :

1. La conception architecturale qui est, depuis longtemps, étudiée surtout au niveau méthodologique est le siège de ce que l'on appelle le "WICKED PROBLEM" <RITEL H. 73 > < SIMON H. 72> , il se caractérise par un certain nombre de propriétés qui font de lui un "problème vicieux":
   - il n'a pas de formulation définitive
   - il n'a pas de règle de terminaison
   - les solutions ne sont pas vraies ou fausses mais bonnes ou mauvaises
   - l'ensemble des solutions potentielles n'est pas énumérable

2. L'espace de résolution du problème de la conception architecturale est par essence un espace graphique < LEBAHAR 80 > ou le graphique est autant un langage de représentation et de communication des informations qu'un lieu de simulation et de résolution. Il paraît, dès lors, difficile de réduire la conception à un graphe ETAT/TRANSITION.

1.2.2 Les systèmes experts généraux

Pour résoudre ces problèmes et rendre "intelligente" l'aide que les ordinateurs peuvent offrir aux concepteurs, des équipes ont pensé à l'utilisation des techniques issues des Systèmes Experts pour l'élaboration de logiciels de CAO <OEY K.H. 86>; les quelques réalisations satisfaisantes, c'est-à-dire qui peuvent être exploitées dans des délais réalistes, traitent de sous-ensembles généralement bien définis : domaines techniques d'aide aux diagnostics thermiques ou de choix de composants face à des critères précis dans un univers limité . Les systèmes experts qui ont été réalisés depuis 10 ans intégrent parfaitement ces approches : aide à la décision, diagnostics ... les généaliser dans des domaines tels que la conception semble très difficile en raison de leur constitution : bases de faits généralement limitées, chaînage arrière (dirigé par les buts), pas de dimension graphique, "explosion combinatoire" pas toujours maîtrisée (dans l'utilisation du chaînage avant, notamment) pour les raisons exposées précédemment.

1.2.3 les sytèmes à raffinement automatique basés sur un système à base de frames

Ces systèmes ont été pour l'essentiel développés autour des techniques à base de frames (plus précisement XRL) que nous analyserons à travers une approche un peu différente dans les chapitres suivants. Il tentent d'affiner le principe de résolution par une technique de raffinement successif d'un macro-objet, que nous appellerons "projet architectural". Cet objet est décrit avec une structure de frames, ses slots sont ensuite raffinés par un "Design Refinement Processeur" de manière automatique. La démarche du processeur de raffinement est basé sur un système à base d'agenda piloté par un sheduler qui a pour but de gérer (à la manière d'un système multitâches) le lancement des différents raffineurs. Cette démarche, qui est à rapprocher de l'utilisation des frames dans un système expert, ne peut fonctionner que si l'on admet la nécessité d'automatiser la démarche de description des objets et si le système est réellement non-continu (c'est à dire non-algorythmique). Cette démarche qui est de loin la plus intéressante et la plus apte à généralisation avec quelques points de convergences avec celle que nous proposons.

Les techniques de recherches heuristiques "classiques" ne fonctionnant pas de manière satisfaisante dans l'espace de "connaissance de la conception", toute avancée de l'intelligence artificielle dans ce domaine doit passer par une étude poussée de la représentation des connaissances en architecture.

1.3 La représentation des connaissances

Ce domaine pose deux questions :
- Quelles sont les catégories conceptuelles nécessaires à l'établissement d'une base de connaissances ?
- Quelles sont les formulations informatiques les plus adéquates pour représenter ces catégories conceptuelles ?

La première question fait l'objet d'une réflexion sur la conceptualisation, en distinguant les différentes entités susceptibles d'intervenir dans notre connaissance du monde et les liens statiques et dynamiques qui associent ces

concepts entre eux (ceci est le fruit d'une étude précise et importante du domaine par les profesionnels de la conception).

La deuxième question, et c'est celle qui nous intéresse dans cette partie, concerne les moyens informatiques mis en oeuvre pour représenter et manipuler ces concepts. Nous partirons, pour réaliser cette étude, des différentes réflexions menées au LI2A et dans d'autres centres de recherche sur la conception et, plus particulièrement, sur les objets manipulés par les architectes en phase de conception, notament la distinction fondamentale entre les concepts abstraits et les objets concrets.

2.UN PREMIER LANGAGE A STRUCTURE DE FRAMES

Nous présentons ici d'abord les spécificités du langage que nous avons implémentées dans la première phase de notre étude ‹CARADANT-85› Les lignes générales se retrouvent dans un certain nombre de langages semblables : FRL ‹ROBERTS-77› et KRL ‹BOROW-77›...

L'idée des schémas, issue d'un article de Minsky ‹MINSKY-75›, est basée sur le fait que nous avons dans notre pensée toute une série de schémas, ou stéréotypes, qui représentent notre passé et l'idée que l'on se fait d'une situation, d'un objet et des gens. Cet à-priori nous aide à réfléchir, à raisonner sur une nouvelle situation. Le terme de frame sous-tend l'existence d'un grand nombre de représentations par liaisons : objet*attribut*valeur. Ils sont les premiers intégrant conjointement : caractère déclaratif (d'où la ressemblance avec un enregistrement dans une base de données) et procédural, par les procédures rattachées à SI-BESOIN et à DEFAUT. La possibilité d'héritage de propriétés est assurée par les propriétés de type SORTE-DE qui permettent de recevoir les propriétés du père.

2.1 L'héritage : la gestion des héritages est assurée par le lien de type "EST-UN" : cette propriété prédéfinie génère un héritage entre les différents objets.

2.2 Le côté déclaratif : aspect "VALEUR" et aspect "DEFAUT"
L'héritage de la valeur par défaut se fait par exploration de la hiérarchie : celle-ci se présente souvent sous la forme d'un reseau. Si l'objet et ses pères n'ont pas de valeur, on va rechercher la première valeur par defaut présente dans la hiérarchie, et cette valeur sera alors la valeur de la propriété pour l'objet.

2.3 Le côté procédural : le côté procédural de la frame est mis en place par le même type de hiérarchie. Il est accroche a certains "aspects" (appeles encore dans ce cas démons) de la structure declarative des objets. Ce sont les modifications opérées sur les propriétés, à la consultation, à l'ajout, ou la modification de valeurs, qui lancent l'execution d'une procédure.

Ce premier prototype nous a permis, sur des exemples architecturaux simples, de soulever les problèmes generés par une approche hiérarchisée des connaissances (déclaratives ou procédurales) :
Si l'approche hierarchisee de type frame convient bien a la spécialisation des concepts: un mur en beton est une spécialisation du concept MUR; un couloir ou un escalier sont des spécialisations du concept de CIRCULATION, il n'en est pas de même pour la notion d'agréga.

2.4 la notion d'"agrégat".
Une pièce est formée de plusieurs entités : murs, fenêtres, portes, planchers, plafond.. qui "héritent" d'un certain nombre d'attributs de cette pièce, mais dans ce cas, nous n'avons plus le concept de spécialisation par rapport à la chambre. La pièce est alors un macro-élément, composé d'objets élémentaires, qui impose un certain nombre de contraintes sur les éléments la constituant ; nous définirons ceci par la notion d'agrégat.

Le concepteur qui manipule à partir d'un certain niveau de définition n'a pas à définir complètement le mur et les différents objets le composant; la machine doit cependant connaître ces différents constituants pour maintenir dynamiquement la cohérence graphique, technique ou économique du projet. La pièce concernée doit alors "prévenir" tous ces constituants de tout changement à son niveau; dans certains cas, le contraire doit pouvoir se faire. Cette approche est celle d'un superviseur qui transmet, à tous les niveaux concernés, les modifications qui ont eu lieu à un certain degré de définition.

Il ne s'agit pas de placer une hiérarchie fictive en terme de nouveaux liens d'héritage de type "PARTIE-DE" gérés par la même structure et ne permettant que des heritages partiels par rapport a une classe générique. En effet, la présence de ce double héritage risque de faire disparaître la notion sémantique de spécialisation, en la substituant à quelque chose de beaucoup plus flou, regroupant la specialisation et l'agregation.

Comment introduire, dans une structure de représentation, la spécialisation et l'agrégation en respectant la sémantique de chacune, en permettant à L'ARCHITECTE, utilisateur final, de transférer ses savoirs et ses méthodes de conception dans cette structure ? Ce probleme qui est sans doute l'un des plus difficiles a résoudre en CAO nous a conduit vers les "langages orientés objets".

3 LE LANGAGE LORCA

Le langage LORCA (Langage Objet pour la Représentation des Connaissances en Architecture) que nous avons défini et implémenté permet de manipuler des structures très puissantes : il allie la puissance déclarative des FRAMES à la modularité des structures de contrôle des langages orientés objets.

Nous avons implémenté dans le langage deux structures de représentation d'objets :
- Un interprète de type SMALLTALK qui permet de gérer les envois de messages entre les différents objets
- Une structure de représentation des données de type FRAMES qui permet de gérer aussi bien le côté déclaratif que le côté procédural.

3.1 le langage SMALLTALK

A1- Toute entité SMALLTALK manipulable est un objet.
Un objet comprend :
- de la mémoire pour ses données
- des méthodes (ou opérations) qui permettent de manipuler ses données

A2- Tout objet est une instance d'une classe
Le principe est de regrouper à l'intérieur d'une classe les objets aux comportements similaires. Une classe est donc une description INTENTIONNELLE d'un ensemble d'objets.

A3- Tout objet est ACTIVE à la réception d'un MESSAGE
La transmission de MESSAGES entre OBJETS est le SEUL moyen de contrôle dont l'utilisateur du langage dispose. Un message peut être vu comme une demande d'actions. Un programme SMALLTALK se résume à l'envoi de messages entre les différents objets.

A4- Toute classe est sous-classe d'une ou plusieurs autres
Toute classe est instance d'une autre classe (la classe initiale étant sa propre mère). Ce principe de HIERARCHIE permet de réaliser le principe d'HERITAGE. Une classe hérite des dictionnaires des méthodes et des champs de ses classes supérieures.

Les langages orientés objets tels que SMALLTALK permettent de représenter, et surtout de manipuler des connaissances, grâce aux points suivants :
- Structuration de la base par différenciation entre classes et instances
- Champs locaux pouvant s'apparenter aux attributs
- Intégration des caractères liés à la représentation et au traitement des connaissances grâce à l'emploi des methodes.

Cependant, il leur manque un certain nombre de caractéristiques que nous avons mentionnées (valeurs par défaut, démons, contraintes), et qui, de ce fait, les rendent inaptes à la représentation des connaissances dans des domaines à fortes tendances inférentielles

L'interprète que nous avons implémenté reprend les travaux qui ont été faits à l'université de PARIS-IV par Pierre COINTE et par Isabelle BORNE : notre modèle n'admet pas le multi-héritage au niveau de la recherche des méthodes accrochées à une classe.

Le générateur de classes est la classe METACLASS qui génère toutes les classes et qui supporte les méthodes que, seules, les classes peuvent réaliser : création d'instances de classe, attachement de variables de classe, de démons

Les objets terminaux ou instances de classe ne peuvent recevoir les méthodes de classe, mais héritent de toutes les méthodes qui ont été définies dans le père de tous les objets du système : l'objet "OBJET".

3.3 La gestion des différentes notions d'héritage

3.3.1 L'heritage des méthodes dans l'interprète SMALLTALK

Les objets sont, comme dans l'ensemble des langages objets, les entités du langage qui contrôlent la réception et l'interprétation d'un message qui leur est envoyé. La recherche de la méthode correspondante commence dans la classe générative de l'objet ; si cette recherche échoue dans cette classe, l'objet receveur continue celle-ci dans l'arbre des surclasses de cette classe. La méthode est interprétée et le résultat est renvoyé à l'objet qui a envoyé ce message.

3.3.1 L'héritage dans la structure de frames

Il faut surtout retenir que les héritages dans les FRAMES permettent la gestion des multi-héritages : une classe peut avoir, pour les gestions évoluées des cohérences, plusieurs classes pères (nous sommes alors en présence d'un reseau).

3.4 Sémantique de quelques instructions intéressantes :

3.4.1 Messages de classe :

- < class CREE objet x1 x2 x3 ... > : Crée un objet instance de la classe "class" avec pour "valeur d'attribut" x1 x2 x3
- < class <- prop valeur > : Donne à la propriété "prop" la valeur "valeur" pour tous les objets de cette classe (si cette propriété n'est pas déjà spécifiée dans les objets terminaux).
- < class ? PROP > : Ramène la variable de classe de la propriété "prop", la recherche est en fait une recherche de type hierarchique dans l'arbre des frames peres de la classe "class".
- <class AT-PROC prop facette valeur> : Sert à générer des attachements procéduraux de type SI-AJOUT, SI-BESOIN, SI-DETRUIT à la classe "class".

3.4.2 Messages pour les objets terminaux

- < objet <- prop valeur> : Permet de réaliser la liaison d'un attribut à une certaine valeur. Avant de faire cette liaison, l'objet regarde dans sa frame (c'est-a-dire dans tout l'heritage, au sens FRAME du terme) si cette valeur est valide. Ce contrôle est assuré par le lien SI-AJOUT.
- <objet ? prop > : Demande à un objet la valeur de l'attribut "prop". Si cette valeur existe dans l'objet, elle est ramenée, sinon la recherche se fait dans la frame de l'objet (ceci permet d'avoir des variables de classe, des valeurs par défaut, ou des valeurs calculées du type SI-BESOIN)

4 EXEMPLE D'IMPLEMENTATION D'UN PROTOTYPE EN LANGAGE LORCA

Cet exemple a pour but de montrer que l'approche objet permet de manipuler, de manière très simple et très cohérente, les différents type d'"héritages" qui sont présents dans le projet architectural. Il ne s'agit pas d'implémenter un niéme L.O mais de montrer comment cette approche est intéressante dans un système de CAO pour permettre la gestion dynamique des contraintes, des valeurs par défaut, et des connaissances de "bas niveaux" des objets du projet architectural.

4.1 Les héritages dûs à l'utilisation d'envois de messages

Nous partons du principe que le concepteur travaille à un certain niveau de définition qui se situe dans un premier temps au niveau du porche. Il manipule, dans ce cas, un objet ayant une sémantique très précise, toute manipulation à

cette "échelle" doit être répercutée au niveau inférieur suivant un certain nombre de règles définies par le concepteur : dans notre exemple, les règles sont des règles de composition.

Nous ne sommes pas en présence d'un objet "porche" spécialisé en d'autres objets, mais devant un concept défini par le savoir-faire de l'architecte comme étant l'assemblage d'objets plus élémentaires suivant un certain nombre de règles.

Prenons un exemple de "méthode" de manipulation : la manipulation de la hauteur du porche.
La hauteur d'un "macro-élément" comme le porche conditionne la hauteur des différents éléments le constituant. Il influence, en effet, la hauteur de la colonnade, de la base, de l'entablement, et du fronton. La méthode PRENDS-HAUTEUR qui est définie dans la classe des porches "PORCHE*" doit donc tenir compte de cela et envoyer des messages aux objets élémentaires pour réaliser ce maintien de cohérence.

```
! PORCHE* APPREND PRENDS-HAUTEUR
[ (hauteur) (RELIE HAUTEUR hauteur)
  (SEND(TOI '? FRONTON )'<- 'HAUTEUR (QUOTIENT hauteur 6))
  (SEND (TOI '? BASE )'<- 'HAUTEUR (QUOTIENT hauteur 6))
  (SEND (TOI '? ENTABLE )'<- 'HAUTEUR (QUOTIENT hauteur 6))
  (SEND (TOI '? COLONNADE )'<- 'HAUTEUR (QUOTIENT hauteur 2))] !
```

C'est la classe PORCHE* qui va permettre à l'objet spécifique PORCHE1, créé par l'utilisateur de manière dynamique durant la conception, de maintenir sa cohérence.
Si PORCHE1 reçoit comme hauteur 400 alors il va envoyer une suite de messages comme : (SEND (TOI '? FRONTON) '<- 'HAUTEUR (QUOTIENT X 6))
Ce message se traduit par : TOI (l'objet qui est le récepteur actuel) se demande quel est son FRONTON et envoie ensuite à cet objet l'ordre de modifier sa hauteur: ici le FRONTON va prendre comme hauteur 400 divisé par 6, donc 66. Les sous-concepts de porche vont donc changer de hauteur quand on a fait une manipulation sur l'objet PORCHE1. On peut généraliser cette démarche en forçant un concept à se modifier lors du changement des valeurs des objets le constituant.

```
! COLONNADE* APPRENDS PRENDS-HAUTEUR
(X)(RELIE HAUTEUR X) (SEND (TOI '? PARTIE-DE) '<- 'HAUTEUR (TIMES X 2))
```

On remarque que la colonnade envoie au concept dont elle fait partie (relié par le lien PARTIE-DE) un message qui le force à modifier sa hauteur. Il est à remarquer aussi que le même message PRENDS-HAUTEUR n'est pas interprété par les deux classes PORCHE* et COLONNADE* de la même manière.

4.2 Les héritages dûs à la spécialisation

Ces héritages sont ceux que nous avons vus grâce à notre premier prototype basé sur une structure de FRAMES. Un objet peut regarder dans ses pères s'il a le droit de prendre une certaine valeur pour une propriété particulière (ou une valeur par défaut) :

```
! COLONNADE* AT-PROC HAUTEUR SI-AJOUT '(VERIF-LIM 100 500 INTERDIT) !
```

Si la valeur de la hauteur de la colonne n'est pas comprise entre 100 et 500, alors l'action est interdite.

5 LES EXTENTIONS DE L.O.R.C.A

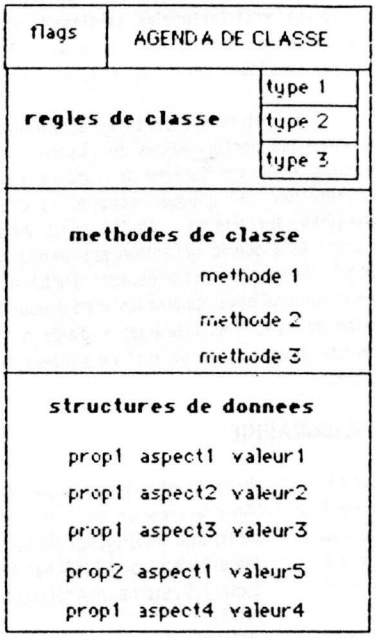
5.1 Vers les systèmes experts

5.1.1 Les différents niveaux d'expertise

LORCA gère pour l'instant ce que nous avons appelé les expertises de bas niveau, à la limite de la description triviale, en tout cas pour un être humain, des objets architecturaux. Cette description doit permettre aux différents objets de prévenir le concepteur en cas de mauvaise utilisation (gestion de cohérence grâce aux frames) mais doit permettre aussi de modifier des éléments qui sont en relation directe : relation de proportion, de positionnement pour des éléments qui ont un certain ordonnancement (fenêtres, soutiens, éléments de rythme). L'introduction d'expertise de niveau plus élevé nécessite une extension de la structure des objets du langage.

5.1.2 les nouveaux objets du langage

Ces nouveaux objets du langage regroupent dans une même structure les objets LORCA actuels, des règles de classe, et une structure de communication plus évoluée. L'idée de base est d'integrer., dans le langage, les règles que l'architecte manipule et qui sont plus que de simples règles de validation de cohérence : ces règles peuvent intégrer des contraintes urbaines, des savoirs thermiques et permettre, à travers des règles de savoir-faire, de déterminer les coûts d'un bâtiment non complètement défini. Le moteur qui manipulera ces règles sera un moteur chaînage avant, largeur d'abord et pourra, soit inférer de nouveaux faits dans la base, soit générer des alarmes au concepteur. Il sera généralement "lancé" par le concepteur quant il estimera cela necéssaire.



Type de règles possibles :

Si la hauteur du bâtiment dépasse 8 mètres, alors le bâtiment change de qualité incendie

Si le bâtiment est de type "immeubles de grandes hauteurs", alors il faut avoir des portes coupe-feu

5.2 Les envois de messages

Les envois de messages pourront se faire de manière nominative (telle qu'ils sont faits dans la plupart des langages orientés objets) ou sous la forme de messages de classe, c'est-à-dire de messages envoyés à une classe qui ne seront interprétés que par les objets qui répondent à un certain nombre de propriétés (telle que cela est fait dans les langages d'acteurs). L'interêt directe de cette démarche est de pouvoir envoyer des messages à des objets non complètements connus mais qui répondent à certains critères. Un gestionnaire de messages sera chargé de prendre les messages collectes dans les agendas de classe et de les distribuer a leurs destinataires. Ce gestionaire sera composé d'un "matcher" capable d'évoluer sur les structures d'objets de LORCA.

5.3 L'interface homme-machine de définition d'objets

L'interface homme-machine qui doit permettre la définition des objets en terme de couples attribut-valeur, et en terme de procedures de maintien de cohérence (par envoi de message entre les objets), sera, comme c'est une tradition dans l'emploi des langages orientés objets, bâtis sur l'approche graphique-souris. Il s'agit ensuite de fournir au concepteur la possibilité de construire, à partir de primitives graphiques de haut niveau, ses propres methodes (procédures). Il faut alors veiller à définir, de manière assez exhaustive, l'ensemble minimum des types d'actions employées : la mise à la proportion et l'allignement... L'éditeur de règles doit aussi utiliser ces potentialites et sera comme le générateur de modèle basé sur éditeur-interprète

L'ensemble de ces extentions est en cours de réalisation : un éditeur graphique piloté par une souris a déjà été connecté au langage, et les nouvelles structures sont en cours d'impiementation.

6 Conclusion

Notre programme futur recouvre un certain nombre de thèmes et axes de recherche, qui concernent aussi bien le champ de l'informatique que celui de l'architecture. Ainsi, bien au-delà de la simple manipulation de traits et de lignes, de "dessins", notre but est de créer un outil capable de gérer des connaissances, des règles et des relations

l'expression graphique d'un "objet" n'est plus alors que l'une de ses nombreuses occurences - certes privilégiée -, à un moment déterminé, dans un contexte précis.

Un système informatique bâti sur l'approche "objets" peut permettre la description de certains types de manipulations architecturales. La classification de ces objets ou actions permettrait l'établissement d'une base de connaissances qui pourrait être vide au départ et, progressivement remplie par l'utilisateur. Ceci lui permettrait de conserver sa propre démarche et concourrait à préserver ses savoir-faire

Enfin, les systèmes experts en architecture sont étroitement liés à la représentation des connaissances qui conditionne les performances du système. Il est donc primordial que la structure choisie pour représenter ces connaissances soit organisée de manière à permettre une exploitation maximale. Ainsi, à chaque niveau de representation, à chaque échelle, à chaque niveau conceptuel, pourraient être associées des procédures d'expertises spécifiques. On pourrait ainsi envisager, dès les premières étapes de l'esquisse, d'aborder des problemes tels que la définition approximative des coûts, l'évaluation progressive du comportement thermique du bâtiment, le respect des prospects d'urbanisme, etc...

Les prochains developpements auront pour but, comme cela a été precisé dans la dernière partie, de permettre la création de systèmes intelligents d'aide à la conception grâce à l'utilisation de techniques qui sont issues des systèmes experts. Ceci permettra d'intégrer une autre dimension à LORCA et de lui donner le rôle d'assistant de conception.

BIBLIOGRAPHIE

Borne I. Micro-Smalltalk pas-à-pas. Draft juin 84.

Borow D. G. "An overview of KRL". In "Cognitive Science", 1977 .

Charniac E. "Artificial Intelligence Programming". Lawrence Erlbraum Associates Publishers, 1980.

Cointe P. Une extension de VLISP par les objets. Science of computer Programming, 161 North Holland 1984

Cullen Y. "Expert Systems in Architectural and Planning Education" Proceeding of the eCAADe, BRUSSEL 1983.

Duplay d. "Méthode illustrée de conception architecturale." Editions du Moniteur 1983.

Dugerdil P Une méthodologie orientee objet pour la représentation des connaissances en C.A.O Architecture G.R.T.C. Marseille 1985.

Goldberg A. "SMALLTALK-80". Addison Welsley Pub, 1983.

Kalay E. Y. Knowledge-Based Computer-Aided Architectural Design Environment. Buffalo N.Y. 1985

Latombe J.C . "Artificial intelligence in computer aided design". in Allen (Ed),CAD systems Amsterdam, North Holland 1979.

LI2A-85 "Les utilitaires et l'intelligence artificielle pour un système d'aide à la conception en Architecture" Rapport final d'une recherche financée par le Secrétariat de la Recherche Architecturale. LI2A Avril 1985.

Caradant D. "Les utilitaires et l'intelligence artificielle pour un système d'aide à la conception en Architecture". Rapport final de recherche. LI2A, Juin 1984.

Michell T.M. "Data Base Management Systems and Expert Systems for CAD. CAD Systems Framework, North Holland Publishing Company IFIP,1983.

Minsky M. "A Framework for Representing Knowledge".in The Psychology of Computer Vision, McGraw-Hill, NY, 1975.

Raphael B. "A computer Program for Semantic Information Retrievial".in Semantic Information Proceedings, M.I.T Press, Cambridge, Mass 1968.

Rittel H. The state of the Art in Design Methods DMG Occasionnal Paper Jan 1972

Roberts R. "The FRL Manual". MIT, AI Report, 1977.

Simon H. The structure of III Structured Problems Articial Intelligence 1973

CARADANT Dominique

Laboratoire d'Informatique Appliquée à L'Architecture.
Ecole d'Architecture de Toulouse
Chemin Aristide Maillol 31100 Toulouse.

Laboratoire des Systèmes Informatiques
Université Paul Sabatier
Route de Narbonne 31400 Toulouse