

## ECCI

### (Environnement de Conception Cognitive en Informatique)

**ECCI** est un environnement de conception et de réalisation des applications informatiques.

La démarche cognitive, les interfaces graphiques et en langage naturel rendent l'utilisation d'**ECCI** (en conception, en validation, en consultation) accessible aux non-spécialistes de l'informatique, experts ou décideurs.

La maîtrise de la complexité étant un objectif central d'**ECCI**, les applications se prêtant le mieux au traitement sous **ECCI** sont :

- modélisation des organisations complexes
- modélisation des systèmes d'information
- modélisation et simulation de workflow
- interrogation en langage naturel de bases de données.

La programmation en logique, la programmation par objets, le raisonnement hypothético-déductif, le réseau sémantique des objets, la résolution de contraintes, l'agenda, les réseaux de transition d'états et de tâches font partie des mécanismes de base d'**ECCI**.

#### *Composants conceptuels.*

Les mécanismes de spécification cognitive d'**ECCI** comprennent :

- A. les **structures** hiérarchiques et sémantiques des objets abstraits
- B. les caractéristiques **descriptives** des objets
- C. l'aspect **comportemental** des objets

- A.**
- la constitution de graphes d'objets
  - la constitution de réseaux sémantiques d'objets
  - la spécification de workflow (tâches, ressources, fournitures, produits)
  - la spécification de réseaux de transition d'états
  - la spécification de réseaux de transition de tâches

- B.**
- les attributs d'objets
  - les méta-concepts
  - les propriétés d'objets (contraintes, réflexes, traits déductifs)

- C.**
- les règles déductives
  - les règles associées aux événements
  - les méthodes et scénarios
  - la communication à travers le réseau des objets
  - les points de vue et les perspectives
  - les contraintes
  - les mondes hypothétiques
  - l'agenda et le tableau noir (paradigmes architecturaux)

### **Base technique.**

*ECCI* se fonde sur la programmation en logique et en hérite les propriétés générales.

En dehors des mécanismes bien connus des environnements orientés-objets (parenté et instanciation multiples, réseaux sémantiques, méta-connaissances, réflexes, méthodes), *ECCI* fait appel aux paradigmes suivants :

- gestion de workflow
- analyse de scènes
- réseaux de transition
- raisonnement hypothético-déductif
- agenda.

La version de base d'*ECCI* est réalisée en **Prolog**.

### **Interfaces.**

Plusieurs modes de communication avec une base de connaissances *ECCI* sont disponibles, y compris les boucles d'interprétation Prolog (logique des prédicats) et *ECCI* (menus déroulants, par type d'entité et d'opération).

Cependant, les modes de communication privilégiés sont graphique et en langage naturel:

- l'interface graphique d'édition de structures, d'objets, de règles
- le langage pseudo-naturel de requêtes et de règles.

Des procédures externes peuvent être utilisées ainsi que tout programme **Prolog**.

### **Maintenance.**

Le matériel utilisable :

- les stations de travail UNIX/Motif (SUN, HP, IBM, Bull)
- les micros en réseau, munis d'émulateur X.
- les PC sous Windows.

*ECCI* offre des mécanismes de sauvegarde :

- des états binaires **Prolog** (illisibles)
- des listings (lisibles par des spécialistes **Prolog**)
- des programmes *ECCI* (lisibles par les concepteurs)

Versions par type d'utilisation :

- *ECCI/Concepteur*
- *ECCI/Réalisateur*
- *ECCI/Utilisateur*
- *ECCI/Runtime*.

### **Côtés originaux d'ECCI.**

- le nombre de paradigmes
- l'équilibre conceptuel entre les aspects structurel, descriptif et comportemental
- l'appel à la programmation en logique
- le langage naturel en tant que langage des requêtes et des règles
- l'accessibilité aux non-spécialistes (par ex. experts et/ou décideurs).

Tous les paradigmes d'*ECCI* étant relativement bien connus en informatique avancée, nous présentons, ci-dessous, les points les plus importants de chacun des paradigmes de base.

#### **Graphe de parenté (relation 'kind\_of').**

- La plupart des systèmes orientés-objets gère un arbre et non un graphe.
- La relation de parenté d'*ECCI* est entièrement dynamique.
- L'héritage multiple s'applique aux liens sémantiques, slots, propriétés, instances, scènes, scénarios, bases de règles, modèles du workflow.
- La fusion de graphes est une opération routinière.
- Les modèles (classes) peuvent être renommés, fusionnés, convertis en instances.
- Des sous-graphes correspondent aux types de règles, scènes, scénarios, modèles du workflow, mondes hypothétiques, catégories grammaticales, types de fenêtres...
- Une centaine de modèles sont prédéfinis (par ex. les modèles du workflow).

#### **Instanciation (relation 'is\_a').**

- Le polymorphisme (attachement multiple d'instances).
- La relation d'instanciation est entièrement dynamique.
- Des instances peuvent être virtuelles (déductibles).
- La fusion de graphes est une opération routinière.
- Les instances peuvent être renommées, fusionnées, converties en modèles.
- Des sous-graphes correspondent aux procédures et tâches (workflow) réussis, interrompus, planifiés ou échoués.
- Les justifications de faits (raisonnement abductif : questions *pourquoi*, *comment*, *où*, *quand*).

#### **Réseau sémantique.**

- Un lien *ECCI* est une relation mathématique (transitivité, (anti-)symétrie, (anti-)réflexivité -> ordres et équivalences).
- Certains liens sont prédéfinis (*composition*, *causalité*, *appartenance*, liens spatio-temporels, *ressource*, *tâche*, *produit* etc.).
- Des liens peuvent être virtuels (déductibles).
- Résolution de contraintes.
- Des sous-réseaux de transitions d'état ou de tâches.

#### **Slots.**

- Des valeurs de slot peuvent être virtuelles (déductibles).
- Résolution de contraintes.
- La transmission de valeurs de slot à travers le réseau sémantique.
- Les propriétés d'entités en tant que slots des méta-modèles.

### **Propriétés de modèles.**

- Contraintes : cardinalité, domaines de modèles.
- Réflexes : *if-attached, if-detached, if-existed*
- Traits déductifs : ensembles d'absorption, stratégie d'héritage, méthodes.
- Traits descriptifs : images graphiques.
- Traits linguistiques : genre, synonymes, rections verbales.

### **Propriétés de liens.**

- Contraintes : cardinalité, domaines de valeurs, demandabilité, accès restreint.
- Réflexes : *if-added, if-needed, if-deleted*
- Traits déductifs : chaînes déductives, hiérarchies de liens, transitivité, (anti-)symétrie, (anti-)réflexivité.
- Traits descriptifs : libellés hiérarchisés.
- Traits linguistiques : rections verbales.

### **Propriétés de slots.**

- Contraintes : cardinalité, types et domaines de valeurs, valeurs par défaut, demandabilité, accès restreint.
- Réflexes : *if-added, if-needed, if-deleted*
- Traits déductifs : canaux de transmission.
- Traits descriptifs : libellés, unités de mesure.
- Traits linguistiques : genre, rections verbales.

### **Propriétés d'instances.**

- Contraintes : domaines de modèles ou de valeurs d'attributs, cardinalités d'attributs.
- Traits descriptifs : images graphiques.
- Traits linguistiques : synonymes.

### **Règle ECCI.**

- La factorisation à travers le concept de base de règles associée aux modèles.
- Le typage de règles.
- Règles déductives et événementielles.
- Déclenchement d'actions des règles à partir d'Agenda.
- Chargement et déchargement de bases de règles.
- Langage pseudo-naturel (*ECCI-Libre*) en tant que langage de règles.

### **Raisonnement hypothético-déductif.**

- Les mondes hypothétiques associés aux hypothèses.
- Les hypothèses sont formulées, acceptées, refutées.
- Les mondes hypothétiques cohérents peuvent être fusionnés.
- Navigation, en parallèle, dans les mondes hypothétiques.
- Démonstrations hypothétiques.
- Versioning.

### **Agenda.**

- Simulation du parallélisme d'exécution de scénarios.
- Propagation événementielle.
- Priorités des tâches sur Agenda.

### **Tableau Noir.**

- L'attente d'événements pour les tâches et les modèles d'état.
- Déclenchement automatique de tâches (scénarios).
- Changement d'état automatique.

### **Workflow.**

- La modélisation de procédures et de tâches.
- Une procédure comme un réseau de transition de tâches (*AND/OR/XOR*).
- Une tâche comme un réseau de ressources, fournitures et produits.
- Implémentation de tâches sous forme de scénarios.
- Modélisation, analyse, simulation et exécution de workflow.
- Procédures et tâches réussis, interrompus, planifiés ou échoués.
- Bases de règles (dispatchers du réseau).

### **Analyse de scènes.**

- Le support du concept de scénario.
- La généralisation des messageries orientées-objets.
- Scénarios en tant qu'implémentation de tâches du workflow..

### **Scène.**

- Rôles et fonctions.
- Graphe de scènes.
- Bases de règles (filtres, scripts, objectifs).

### **Scénario.**

- Filtres, scripts et objectifs.
- Acteurs.
- Déclenchement de scénarios à travers l'Agenda.
- Scénarios en attente d'événements (Tableau Noir).

### ***Langage pseudo-naturel, ECCI-Libre.***

- langages de requêtes et de règles
- phrases coordonnées et subordonnées
- négations syntaxiques et sémantiques
- références d'objet et qualificatifs
- réactions verbales associées aux entités
- analyseurs non-déterministes

### ***Bibliothèques ECCI.***

- 500 primitives.
- Références de niveaux conceptuels. Points de vue.
- Audit des procédures ou modules. Statistiques, documentation.
- Interfaces de programmation en Prolog, C, Java.

### ***Editeurs.***

- Structures.
- Entités.
- Règles.
- Documents (textes, programmes *ECCI*, images, sons).

### ***Modes de communication.***

- Le langage pseudo-naturel (*ECCI-Libre*).
- Éditeur de structures.
- Prolog.
- Menus déroulants.

### ***Architecture.***

- 100 modules.
- Base de connaissances : bases de faits et bases de règles.
- Niveau conceptuel (*meta-meta, meta, user, application*).
- Univers et réalisation.
- Graphe de mondes hypothétiques.
- Agenda et Tableau Noir.
- Sauvegardes et chargements de programmes *ECCI*.

### **Limites physiques.**

- Le nombre d'objets est pratiquement illimité (virtualité).
- Le nombre de règles est pratiquement illimité (déchargement dynamique de bases de règles).

### **Maintenance des applications.**

- Génération de programmes [ECCI](#).
- Fusion d'applications.
- Audit de bases de connaissances.
- Aide en ligne.

### **Documentation.**

- Le Guide de l'Utilisateur.
- Le Manuel de Référence [ECCI](#).
- Le Manuel de Référence Prolog.
- Les démos "pédagogiques" : *ec\_paradigms*, *ec\_primitives*.
- Résumés des paradigmes.

### **Publications.**

**ILINE, H.**

*Un environnement orienté-objets en Prolog,*  
CIAM - 86, MARSEILLE, France

**ILINE, H. & KANOUI, H.**

*From Logic Programming to Object Programming,*  
IJCAI - 87, MILAN, Italy

**GIRAUD, C. & ILINE, H.**

*Object-Oriented Paradigms in Intelligent CAD,*  
CAD and AI - 88, CAMBRIDGE, UK.

**ILINE, H.**

*Techniciens vs Cognitivistes,*  
Journées d'AVIGNON - 91, AVIGNON, France

**ROCHEGUDE C. & ILINE, H.**

*La Modélisation des Organisations Complexes,*  
IA-95, MONTPELLIER, France

**ILINE, H.**

*Le Langage Naturel dans le Contexte des Bases de Connaissances,*  
IA-95, MONTPELLIER, France

**ILINE, H.**

*Natural Language Query of Knowledge Bases,*  
PAP-96, LONDON, UK