Construction itérative de bases de connaissances descriptives et classificatoires avec la plate-forme à objets IKBS. Application à la systématique des coraux des Mascareignes

David Grosser

Université de La Réunion

Institut de Recherche en Mathématiques et Informatique Appliquées

Construction itérative de bases de connaissances descriptives et classificatoires avec la plate-forme à objets $\mathcal{IKBS}$.

« Application à la systématique des coraux des Mascareignes »

THÈSE

présentée et soutenue publiquement le 1 Février 2002

pour obtenir le grade de

Docteur de l'Université de La Réunion

Spécialité Informatique

par

David Grosser

Composition du jury

Rapporteurs :	Jean-François Perrot	LIP6 (Paris)
	Amedeo Napoli	CRIN (Nancy)
Examinateurs :	Edwin Diday	Lise-Ceremade (Paris)
	Michel Pichon	EPHE (Perpignan)
	Gérard Faure	ISIM (Montpellier)
	Régine Vignes-Lebbe	LIS (Paris)
Directeurs de thèse :	Henri Ralambondrainy	IREMIA (La Réunion)
	Noël Conruyt	IREMIA (La Réunion)

Mis en page avec la classe thloria.

A mon adorable Sophie qui n'a que deux ans et demi, mais qui n'est pas moins sage pour autant...

Table des matières

Table des figures

Introduction

1 La machine *pensante* ...

« *L'homme peut vivre dans un château fort, une ferme, une caravane, un igloo, une hutte ou bien une tente* ».

1. Quel sens peut accorder une machine à cette phrase ?
2. Comment l'entendons nous nous-même ?
3. Comment pouvons nous lui transmettre ce qui est implicite pour nous ?

Notre mémoire est riche d'images mentales, de sens cachés, de prototypes associés aux mots. Notre formidable machine cérébrale crée des liens et des structures entre les mots et les sens, qu'il faut expliciter et lui décrire précisément, si nous souhaitons un jour que la machine nous comprenne et qu'elle puisse, à son tour, retransmettre ce qu'elle a appris de nous.

Nous pouvons pour cela chercher un terme générique, mais suffisamment précis qui englobe chacune des habitations évoquées et qui les désigne toutes. Cela n'est pas toujours évident ; le mot *habitation* peut convenir, mais il est un peu trop général. Il existe en effet un grand nombre d'habitations qui ne sont pas concernées par la phrase. Le mot *maison* n'est pas assez général, une tente n'est pas une maison.

Trouver un mot pour désigner un ensemble d'exemples, un terme générique, c'est trouver un concept qui les contient tous. Ceci peut être réalisé si l'on connaît les caractéristiques de chaque habitation et analyser celles qui sont communes. Nous pouvons, à l'aide de **connaissances descriptives** sur les objets étudiés, distinguer des sous-groupes homogènes qui expriment des catégories intermédiaires comme : les châteaux et les fermes sont en pierre, les huttes en paille, les autres en caoutchouc et en fer ou encore en glace. Les unes sont transportables, les autres immobilisées à jamais. Il est plus confortable de vivre dans un château fort que dans une tente, etc. Et ainsi nous approcher peut être un peu plus du concept recherché, ou bien éventuellement le créer s'il ne préexiste pas.

Dans notre compréhension, nous ordonnons également les mots et les sens, nous établissons des regroupements, selon différents critères. Dans notre exemple, ces critères peuvent être les matériaux utilisés pour la construction, la mobilité ou l'immobilité des édifices. Ou encore l'ordre induit par la notion de confort. Ces résultats nous permettent de mieux appréhender la réalité, les mots qui servent à l'exprimer et le degré de généralité des différents concepts mis en jeu. En effet, ils établissent des relations que nous utilisons pour désigner que tel mot est plus général que tel autre, et forment ainsi des connaissances **classificatoire**.

L'utilisation intelligente par la machine des organisations que nous interprétons dans la réalité et que nous lui transmettons, peuvent nous aider à mieux appréhender, nommer et connaître cette réalité qui nous échappe souvent, par sa subtilité et sa complexité. Elle peut par exemple nous aider à déterminer le concept (la classe) auquel un objet particulier peut appartenir. Cette opération est appelée **identification** ou classement. Elle peut également nous aider à créer des classes, les nommer, les caractériser et les organiser selon leur degré de généralité. Ce que l'on appelle **classification** ou inférence de taxonomies.

1.1 . . . dans notre société

De plus en plus, dans notre société industrialisée, un usage très important est fait des ordinateurs dans le but de nous aider à prendre des décisions. La machine « pensante » est entrée au coeur de notre mode de vie, en particulier grâce à sa formidable capacité à communiquer [1], à calculer vite [2] et à représenter et manipuler des connaissances.

En effet, toute entreprise moderne doit se donner les moyens de capitaliser ses connaissances pour rester compétitive. A l'heure actuelle, l'avantage concurrentiel d'une firme repose sur son portefeuille de savoirs et savoir faire. Ce qui donne à une entreprise un avantage décisif sur ses concurrents est le fait de posséder un actif qui lui permette de se différencier sur un marché. Mais pour que cet avantage soit durable, il faut que cet actif ne soit pas facilement appropriable, accessible et imitable par d'autres. Les ressources les plus importantes pour les organisations ne sont donc ni la terre, le travail ou le capital. Il s'agit des actifs intangibles spécifiques à une firme, c'est-à-dire, ses connaissances explicites (ou transcrites) mais surtout tacites [3].

Cette gestion des savoirs se fonde principalement sur de grandes bases de données associées à des outils d'analyse pour l'aide à la décision. L'augmentation croissante et toujours plus rapide de la quantité d'information disponible ont amené le développement de techniques sophistiquées pour explorer les données [4], en tirer l'information pertinente, la « substantifique moëlle » [5] afin de nous aider à prendre les bonnes décisions [6] pour ainsi évoluer progressivement de la société de l'information à une société fondée sur les connaissances.

1.2 . . . pour les objets de la Nature

La plupart des données manipulées dans les domaines industriels se prêtent bien à une représentation sous forme de tableaux de données (ou tables relationnelles) pour décrire des objets *manufacturés* ; Dans les Sciences de la vie, représenter, analyser et transmettre les objets naturels est beaucoup plus difficile. Ceux-ci adoptent en effet des formes souvent complexes, délicates à observer, variées, imprécises et évolutives. Aussi, décrire ces objets demande beaucoup de temps et d'expérience ; Les classer nécessite une connaissance étendue du domaine, l'exception étant le

1. La « toile » a fini d'être tissée tout autour de la terre.

2. L'accès à des machines personnelles d'une puissance de calcul de 1GHz (1 milliard d'opérations à la seconde) est désormais chose courante.

3. D'après un article de Jean-Claude Tarondeau, « Le management des savoirs ».

4. *Découverte de Connaissances dans les Bases de Données* ou *Knowledge Discovery in Databases*.

5. *fouille de données* ou *Data Mining*.

6. Les disciplines sont ici très variées : *Analyse de Données* (AD), *Raisonnement à Partir de Cas* (RàPC ou CBR), *Apprentissage à partir d'Exemples* (Machine Learning), *Intelligence Artificielle*, etc..

plus souvent la règle.

Or, de tout temps, les scientifiques et les observateurs de la Nature se sont intéressés à représenter, identifier et classifier les organismes vivants, afin de mieux la connaître, la comprendre, la protéger ou l'exploiter pour son propre intérêt.

Depuis le naturaliste suédois Linnaeus (1707-1778), qui a défini la nomenclature binominale, de vastes systèmes classificatoires ont été construits qui ont permis à des générations de scientifiques d'améliorer la classification des organismes et d'identifier les spécimens sur la base des organisations taxonomiques. La science de cette organisation de la Nature est appelée « la Systématique ».

2 La Systématique

> « *La Systématique est l'étude et la description de la diversité des êtres vivants, la recherche de la nature et des causes de leurs différences et de leurs ressemblances, la mise en évidence des relations de parenté existant entre eux, et l'élaboration d'une classification traduisant ces liens de parenté* »(Matile et al. 1987).

Régines Vignes (2000) souligne l'importance de la Systématique pour l'étude de la diversité : elle est à la fois un préalable pour toutes les recherches en biologie pour lesquelles l'*identification* précise et la position dans la classification des objets d'étude est indispensable, ainsi qu'un aboutissement, car elle représente une connaissance synthétique sur les êtres vivants.

> « Il est en effet déraisonnable d'élaborer des hypothèses scientifiques sur les molécules, les cellules, sur les processus évolutifs et les parentés entre organismes, en se privant des connaissances sur les organismes actuels et fossiles. La Systématique apparaît donc comme le dénominateur commun aux diverses approches de la biologie comparative et évolutive, fournissant par la classification un cadre de référence universel pour l'étude du vivant. ».

La Systématique est une science vaste et complexe ; si elle étudie essentiellement les niveaux d'intégration allant des organismes aux taxons, qui constituent le coeur de la discipline et que l'on nomme la *Taxonomie*, elle fait par contre appel à des données provenant de tous les niveaux d'intégration, des données moléculaires aux données écologiques (figure 1).

2.1 La démarche expérimentale des systématiciens

Tout au long de sa recherche, le systématicien apprend à reconnaître les espèces à partir d'un travail de bibliographie dans les muséums, d'observations effectuées en laboratoire sur des échantillons et d'échanges d'informations avec d'autres chercheurs. Il construit progressivement un modèle de description de son domaine qu'il applique à de nouvelles observations. Les erreurs d'identification l'amènent à remettre en cause son propre modèle. Il acquiert ainsi petit à petit une intuition du domaine, qui l'élève progressivement au rang d'expert reconnu par la communauté scientifique. Ce cheminement procède par un aller-retour, une **itération** entre les

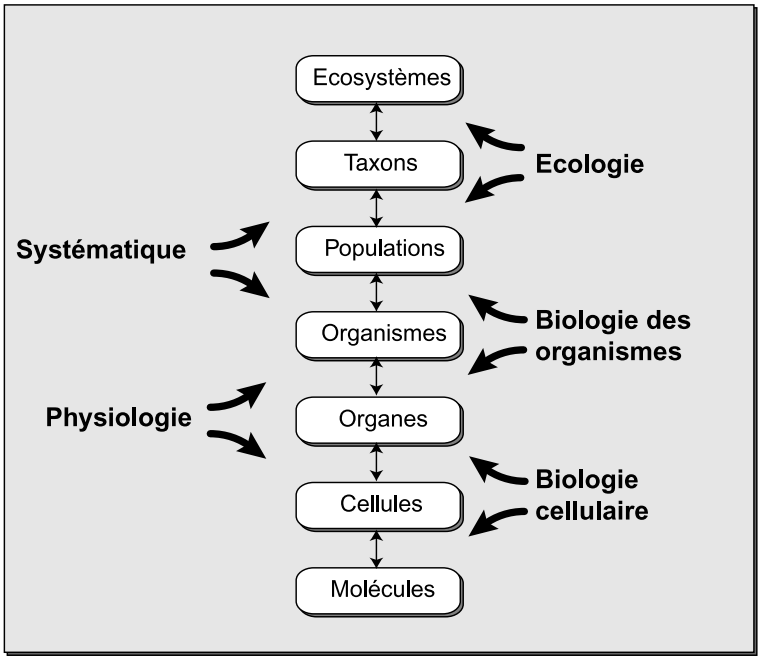


FIG. 1 – *Place de la Systématique dans la Biologie d'après (Lebbe 1995, Vignes 2000)*

informations reçues, son savoir-faire et les modèles bâtis pour représenter cette connaissance descriptive. En effet, l'expérience se nourrit de l'apport des nouvelles informations qui évoluent au fur et à mesure de l'amélioration des techniques d'observation (observations morphologiques au microscope, caractères biochimiques, génétiques, etc.).

La Systématique est également une discipline de terrain, car elle est fondée sur l'étude de spécimens récoltés dans la nature. Pour certains groupes zoologiques, il peut être très difficile de réunir ces échantillons ; le systématicien doit parfois se faire explorateur, voyager dans de lointains pays ou chercher l'organisme rarissime caché au fond des océans. Ces hommes d'expérience sont des experts, spécialistes d'un groupe zoologique particulier, d'une certaine branche de la Taxonomie. Ils sont les seuls qui puissent identifier avec quasi-certitude un échantillon, lui donner un nom d'espèce qui permettra de déduire les propriétés (pharmaceutiques, toxicologiques) et le rôle écologique de l'organisme au sein du biotope. C'est pourquoi, ils sont donc des personnes ressources très recherchées par une partie importante de la communauté scientifique et économique.

2.2 Problèmes de la Systématique

Malheureusement, dans de nombreux domaines de la Systématique (en particulier ceux qui ne concernent pas directement d'applications immédiates et donc commerciales), la connaissance déjà insuffisante, se perd peu à peu. Ce problème, déjà ancien et décrié à plusieurs reprises lors de grands congrès internationaux[7] ne s'est pas amélioré depuis, et les quelques trop rares spécialistes sont sur le point de partir à la retraite.

7. par exemple lors de la conférence de Guam en 74 : *1st International Symposium on Corals and Coral Reefs at Mandapam Camp.*

En biologie marine

Ce problème est particulièrement vrai en biologie marine, comme en témoignait déjà Lévi et al. (1978) :

> « *L'identification des Invertébrés marins des eaux tropicales est une opération longue et délicate. Elle se heurte, en effet à trois obstacles majeurs qu'il faut peu à peu surmonter. En premier lieu, l'absence ou la rareté de spécialistes qualifiés, qui puissent disposer de surcroît, du temps et de l'intérêt nécessaire pour participer à ce travail. Il faut par exemple savoir qu'il existe au plus, actuellement dans le monde, trois zoologistes capables d'identifier des animaux aussi répandus que les Gorgones, les Alcyons ou les Eponges des régions tropicales, et la situation n'est pas meilleure pour beaucoup d'autres groupes d'animaux et pour les algues marines. Le deuxième obstacle est la pauvreté relative des données actuellement acquises sur la faune marine tropicale Indo-pacifique... Seules quelques zones très limitées, au Sud de l'Inde, à Madagascar ou en Australie ont fait l'objet d'études systématiques prolongées sur le terrain... Enfin, la troisième difficulté résulte de la nature même du travail d'identification qui, en l'absence d'ouvrages de détermination spécialisés, déjà rédigés, exige concurramment le recours à l'examen des échantillons types dispersés dans divers musées et à l'étude, in vivo, de la variabilité des populations de chaque espèce...* ».

Pour les coraux des Mascareignes

Pour les coraux, un très petit nombre d'experts se partagent la connaissance dans le monde. Dans quelques années, l'expertise accumulée pour les identifier ne sera plus disponible. En ce qui concerne l'archipel des Mascareignes par exemple, il ne restera plus que les 3000 spécimens de la collection Faure (1982), répartis au Muséum National d'Histoire Naturelle de Paris, au Muséum d'Histoire Naturelle de St-Denis de la Réunion et à l'Université de la Réunion, et les monographies pour reconnaître les espèces.

Connaissant le rôle très important des écosystèmes récifaux pour l'équilibre de la planète et les dégradations continuelles opérées par l'homme sur son milieu naturel, il est impératif de pouvoir transmettre l'expérience des experts aux personnes chargées de l'étude de la biodiversité des récifs. A l'île de la Réunion, le lagon est en voie de perdre une grande partie des espèces qui étaient courantes il y a dix ans à peine.

Les systématiciens n'ont aujourd'hui malheureusement pas les moyens de mener à bien des études de grande envergure et sont incapables de faire face à la demande sociale internationale ; les réponses ne sont réalistes qu'en faisant appel à l'automatisation de certaines tâches. En effet, face à ces difficultés, de par l'ampleur et la complexité des informations à gérer, l'informatisation des données et des connaissances de la Systématique (données de collection, descriptions, classifications) apparait comme un moyen privilégié et même une nécessité pour atteindre ses objectifs.

2.3 Informatique et Systématique

Bien que depuis quelques années, l'informatisation des données et des connaissances utilisées et produites par l'étude de la biodiversité apparaisse comme une nécessité, souvent même comme une urgence, les grandes réalisations d'informatisation en Systématique sont relativement peu nombreuses en regard des grands projets internationaux qui foisonnent (Bailly & Lebbe 1996).

Ces grands projets internationaux visent à répondre à trois besoins principaux :

1. Une demande des systématiciens eux-mêmes : besoin de recherche méthodologique, besoin de meilleurs accès aux données telles que les collections de référence, etc.
2. Une demande des autres scientifiques : besoin de classification, d'identification et problèmes de nomenclature.
3. Une demande de la société en général : pour la gestion de l'environnement, les lois de protection et de gestion de la biodiversité nécessitent d'avoir des noms ayant un sens stable et des experts avec les compétences d'identification.

De nombreuses bases de données scientifiques ont été développées et diffusées ces dernières années sur CD-ROM ou sont accessibles par le réseau Internet : *World Biodiversity database* de l'ETI en Hollande, *Reefbase* et *Fishbase* de l'ICLARM aux Philippines, *Coral Id* à l'AIMS, etc.[8]. Nous pouvons considérer que ce sont des *systèmes d'information* taxonomique et biogéographique concernant un petit domaine de la Systématique comme certains groupes zoologiques. Des projets mondiaux de plus grande envergure sont en cours de réalisation, comme par exemple *Systematic Agenda 2000* dont les objectifs sont de réaliser l'inventaire et la description de tous les êtres vivants, de construire des classifications traduisant l'histoire de la vie et de gérer et diffuser efficacement l'information ainsi constituée.

2.4 Bases de données ou bases de connaissances ?

En fait, ces systèmes reproduisent généralement sous une forme électronique (base de données), ce qui existe déjà de manière écrite sous la forme d'articles et de monographies. Celles-ci traitent des descriptions de taxons, de clés diagnostiques pour l'identification, etc. Ces approches sont adaptées dès lors que les taxons sont précisément explicités et stables, ce qui est le cas en botanique par exemple, mais cette informatisation des données est insuffisante dès lors que la connaissance évolue rapidement, que les taxons et les caractères diagnostiques sont remis en question fréquement et qu'aucun consensus existe entre les experts. Ce qui est le cas dans le domaine de la biologie marine (coraux, éponges, hydraires, etc.).

Trop souvent, la réalisation de système d'information et de bases de données est perçue par les systématiciens comme relevant seulement de la technique, comme une transcription de l'information explicitée dans les monographies. Cela néglige complètement d'une part l'étape de modélisation indispensable pour représenter non pas uniquement les mots mais également leur sens et leur inter-relations (Vignes 2000), et d'autre part, les différents modèles de *représentation des connaissances* développés depuis les années 70, par les chercheurs en Intelligence Artificielle.

8. Nous présentons à l'annexe E, quelques adresses de sites Internet relatifs au domaine.

Ceci, dans le but de bénéficier de toute la puissance des ordinateurs, non seulement pour transmettre et diffuser l'information, mais également pour raisonner sur les connaissances modélisées.

En effet, de nombreux modèles généraux de représentation des connaissances ont été élaborés et appliqués avec succès dans des domaines très variés (pour la médecine, la biologie moléculaire, la génétique, etc.), mais très peu de systèmes spécifiques n'existent à l'heure actuelle permettant d'exprimer toute la richesse et la nuance des concepts manipulés en Systématique [9].

Dans ce contexte, nous pensons donc qu'il est fondamental que les chercheurs en systématique puissent aujourd'hui avoir accès à des solutions informatiques qui s'appuient sur des modèles et des technologies plus puissants de gestion et de représentation des connaissances, dans le but de réaliser de véritables « *Ontologies des connaissances d'un domaine de la Systématique* ».

2.5 Ontologie

Les *ontologies* d'un domaine peuvent être vues comme le moyen d'établir une base conceptuelle commune, dans le but de communiquer des connaissances relatives au domaine, pour réaliser différents objectifs. Ces dernières années ont vu un développement considérable des techniques de découverte de connaissances par l'intermédiaire de la création, automatique ou semi-automatique, de structures relationnelles complexes de connaissances. Par exemple, un courant de recherche actuel très actif vise à construire un *Web sémantique* [10] à partir de données structurées ou semi-structurées (XML, HTML, bases de données, etc.) ; à capturer la sémantique associée aux mots à partir de texte (extraction de connaissances à partir de textes ou *Text Mining*), ou encore à développer des mécanismes sophistiqués de recherche d'information, à l'aide d'agents intelligents explorant le Web (Systèmes Multi-Agents).

En Systématique, réaliser l'ontologie (ou les ontologies) d'un domaine consiste à définir la terminologie employée (thésaurus), préciser les concepts et les mots utilisées, et cibler les objets de l'étude (les collection, la zone géographique concernée, etc.). Ce qui peut être réalisé à l'aide des Systèmes de Gestion de Bases de Données et des Systèmes d'Information Géographique. Mais nous pensons qu'il faut aller plus loin.

L'approche actuelle fondée uniquement sur l'informatisation des données de la systématique, généralement à l'aide des bases de données relationnelles est insuffisante. Bien sûr, celles-ci offrent de puissants moyens pour stocker un grand nombre d'informations et y accéder par l'intermédiaire de requêtes, mais elles présentent un certain nombre de limitations, comme nous l'avons souligné dans Rousse et al. (2000).

2.6 Limites des bases de données relationnelles

A l'heure actuel, les Systèmes de Gestion de Bases de Données Relationnelles (SGBDR) sont largement utilisées pour assurer le stockage (persistance) et la diffusion des données. Les entités sont représentées par un ensemble de *tuples*, organisés en tables inter-reliées à l'aide d'index. Les tables peuvent être consultées ou générées dynamiquement (on appelle *vues* ces tables virtelles), par l'intermédiaire de requètes, généralement exprimées dans le langage standard SQL.

9. L'exposé des besoins précis de la Systématique en matière de représentation des connaissances, ainsi les différents modèles informatiques feront l'objet d'une étude détaillée au chapitre suivant.

10. Cf. annexe E pour une liste d'adresses de sites Internet concernés par le domaine.

Pour notre problématique, le modèle relationnel souffre de limitations dues principalement au fait que :

- Aucune sémantique n'est associée aux données internes ;
- Aucune sémantique n'est associée aux relations internes et externes à la base de données ;
- Il est difficile de faire évoluer le modèle de données, fixé *a priori*, avant l'insertion des données ;
- Pas de représentation hiérarchique des données.

Rousse (2001) propose un modèle de représentation des « objets-métiers de la Systématique » fondé sur les Systèmes de Bases de Données Orientées-Objets (SGBDOO) pour tenter de pallier à ces limitations. Le modèle développé, appelé OSIS [11] permet d'associer du sens aux données, aux relations et autorise plus de souplesse dans l'accès aux données taxonomiques, par la possibilité d'observer les données sous de multiples *points de vue*.

2.7 Vers les Bases de Connaissances (BC)

Cependant, quel que soit le système d'information utilisé (SGBDR ou SGBDOO), le recherche d'information associée à un spécimen d'étude, nécessite de connaître précisément son nom (ou taxon), qui sert généralement de clé d'accès à l'information. Or, trouver ce nom est précisément la tâche du spécialiste et représente toute la problématique de l'identification : un expert n'est pas simplement une encyclopédie vivante qui connaît par cœur un ensemble d'informations, c'est aussi une personne qui sait raisonner sur un domaine particulier pour prendre les bonnes décisions (identifier une maladie ou une espèce par exemple). Nous considérons donc que les systèmes d'information *bibliographique*, *taxinomique* ou *géographique* [12] sont de puissants outils pour le stockage (ou persistance) des données et des informations (savoir encyclopédique), mais sont inadaptés pour représenter des connaissances d'ordre stratégique, de type savoir-faire ou des raisonnements plus évolués.

Les « Bases de Connaissances » sont une réponse à ces limitations. Appliquées aux sciences de la vie, elles offrent une aide à l'identification pour ceux qui ne connaissent pas ce nom, mais également une aide à la classification pour les spécialistes dont le domaine n'est que partiellement étudié (ce qui est vrai pour une grande partie de la faune marine, ainsi que pour la flore et la faune tropicales terrestres). Elle offre la possibilité d'élaborer une véritable « Ontologie d'un domaine » de la Systématique, à travers une modélisation fine des concepts, des descriptions, de la terminologie utilisée et du sens associé aux mots.

3 Les Systèmes de Gestion de Bases de Connaissances

Les bases de connaissances sont difficilement dissociables des systèmes utilisés pour les développer, du fait de la spécificité des formalismes utilisés, par rapport au modèle classique sous

11. **OSIS** pour *Object-oriented Systematics Information System*
12. **SIG** : Systèmes d'Information Géographique.

forme de tableaux de données. Afin de clarifier le discours, nous introduisons les différentes notions (données, connaissance, systèmes d'information, etc.) permettant de mieux appréhender le concept de *Bases de Connaissances* (BC) et les Systèmes de Gestion utilisés [13] pour les construire.

3.1 Données, information et connaissance

La recherche de la connaissance est une partie importante de l'activité humaine, en tout cas l'un de ses moteurs. La connaissance revêt un double aspect : d'une part l'accumulation de données ; d'autre part celui d'une compréhension des phénomènes observés, résultats des relations entre les faits connus et de la vérification d'hypothèses explicatives par la recherche de nouvelles données [14].

Par **données**, on entend des faits élémentaires enregistrés sur des phénomènes du monde extérieur et qui peuvent être considérés comme acquis (ou presque). Le mot phénomène est compris au sens large.

L'**information** peut être considérée comme l'augmentation de la signification qui peut être apprise à partir d'un ensemble de données, de cette « matière brute » que constituent les faits (Langefors 1977). La distinction entre données, informations et connaissances apparaît ainsi progressivement : la manipulation répétée des données permet de tirer des hypothèses que l'on cherche ensuite à mettre à l'épreuve sur des faits nouveaux. Ces hypothèses sont des informations qui peuvent se renforcer et prendre du sens pour l'analyste. Elles peuvent aussi échouer par le fait de contre-exemples. La synthèse des résultats d'observation se transforme alors en connaissances pour l'aide à la décision.

La **connaissance** est donc la résultante des ajouts successifs d'informations aux données de départ, informations qui sont elles-mêmes considérées comme des données pour les inférences de niveau supérieur. Un axiome ou un théorème peut être considéré comme une connaissance en mathématique ou en logique. Un processus intelligent est un processus qui a la faculté de connaître et de comprendre, c'est-à-dire de manipuler des données et d'inférer de nouvelles connaissances (Kayser 1984).

3.2 Les systèmes d'information

Joël Rosnay (1975) (« le Macroscope ») définit un *système* comme :

> « *Un ensemble d'éléments en interaction dynamique, organisés en fonction d'un but. L'analyse des systèmes consiste à définir les limites du système, à modéliser, à identifier les éléments importants et les types d'interaction entre ces éléments, puis à déterminer les liaisons qui les intègrent en un tout organisé* ».

Pour Planche (1988), un système d'information est

> « *un système qui a pour objectifs de rassembler, de traiter, de manipuler et de fournir les informations nécessaires à certaines activités.* »

13. Systèmes de Gestion de Bases de Connaissances (SGBC) ou Systèmes à Base de Connaissances (SBC)
14. Ce que l'on pourrait appeler **érudition** d'une part et **expérience** d'autre part.

Il peut donc comporter des éléments manuels et des éléments informatisés. Un modèle de système d'information est une représentation d'un système, cherchant à en faciliter la compréhension. Un modèle est nécessairement une simplification de la réalité, une abstraction cachant certains détails pour en mettre en valeur d'autres. Pour Tardieu et al. (1984) la construction d'un système d'information se situera en permanence entre :

- l'analyse de l'organisation et du système d'information perçu comme un objet naturel,
- la conception d'un *objet artificiel* ayant pour but de représenter cette organisation pour en améliorer l'efficacité.

3.3 Les modèles de représentation des connaissances

Le mode habituel et le plus ancien de représentation et de transmission des connaissances est le langage naturel. Ce n'est pas le seul. Depuis longtemps les hommes fixent leurs connaissances sur des supports divers (pierre, papier) en utilisant des techniques variées (écriture, dessin). En lui-même le langage n'est pas la connaissance mais une suite de phonèmes ou de caractères, c'est-à-dire seulement un mode de représentation des données et des connaissances, une notation. Pour comprendre cette représentation il faut pouvoir associer un sens à chacun des termes utilisés, les symboles et à leur combinaison, c'est-à-dire disposer d'un *modèle de représentation des connaissances*. De nombreux modèles spécifiques ont été développés en fonction des besoins propres. Ainsi, une carte de géographie utilise un modèle graphique de représentation des connaissances géographiques. La notation mathématique est fondée sur un modèle de représentation des concepts mathématiques, etc.

L'informatique et plus particulièrement les langages de programmation ont longtemps engendré une séparation nette entre les données enregistrées et leur signification (Tsichritzis & Lochovsky 1982). Par exemple l'expression en langage naturel "Pierre est brun et mesure 1,74 m" est interprétée comme "la couleur des cheveux de Pierre (qui est une personne de sexe masculin) est brune et sa taille est de 1 mètre et 74 centimètres". Les concepts de personne, de cheveux, de couleur de cheveux, de taille sont implicites pour l'être humain. Au contraire, un programme informatique pourra stocker le chiffre 1,74 et les chaînes de caractères "Pierre" et "brun" sans se soucier de la signification de ces termes. Les concepts utilisés ne sont pas *a priori* reliés dans le programme. La réalisation d'un modèle de données ou de connaissances essaie de combler ce fossé en associant aux données enregistrées un modèle de représentation puis de signification des données. Elle représente une évolution naturelle de l'informatique vers l'ingénierie ou la *gestion des connaissances*.

Modèles généraux et modèles spécifiques

Chaque utilisateur d'un système informatique utilise habituellement un modèle spécifique, propre à ses besoins. La réalisation d'un modèle général vise à coordonner et à intégrer des modèles plus spécifiques. L'objectif du concepteur d'un système informatique est de définir le modèle le plus général possible.

La puissance d'un modèle général peut être observée par sa capacité d'intégration de modèles plus spécifiques de la réalité telle qu'elle est vue ou sa puissance de représentation de la réalité

telle qu'elle existe. De même qu'une décision politique se réfère à un modèle politique, une décision informatique se réfère à un modèle de données et/ou de connaissances. En l'absence de modèle, les décisions deviennent aléatoires (Degoulet 1998).

3.4 Les systèmes à base de connaissances

Un *système à base de connaissances* est un programme capable d'accomplir une tâche "intelligente". Il est généralement constitué d'une *base de connaissances* relative à un domaine d'application et un ensemble de *mécanismes d'inférences* qui manipule cette base pour résoudre un problème donné (Napoli 1992). La base de connaissances contient des faits ou données brutes caractérisant les objets du domaine considéré : les règles permettent de manipuler ces faits, ainsi que des heuristiques et des stratégies de raisonnement exprimant la façon de se servir des règles. Les opérations de manipulation des connaissances, de recherche et de raisonnement, sont indissociables de la représentation elle-même.

Ainsi, élaborer une base de connaissance nécessite d'adopter un formalisme de représentation des connaissance (Kayser 1984, Levesque 1986), c'est-à-dire des structures de données appropriées : au stockage (SGBD), à la manipulation d'informations (SI) mais aussi à l'inférence de nouvelles connaissances (SGBC).

Pour formalisation des connaissances il est nécessaire au préalable de précisément les différents niveaux de représentation des connaissances.

3.5 Niveaux de représentation des connaissances

D'après Reichgelt et al. (1989), nous pouvons distinguer quatre niveaux de représentation des connaissance. Ils sont classés de la façon suivante dans l'ordre dans lequel ils doivent être successivement abordées :

- le niveau conceptuel, appelé **sémantique**. Ce niveau définit les liaisons entre les primitives du modèles,
- l'épistémologie, appelé **syntaxique** qui définit les primitives du langage,
- la logique utilisée, appelé **pragmatique**,
- le choix d'**implantation** informatique.

En sciences de la vie, comme le soulignent (Lebbe 1991, Vignes 2000), le problème de la représentation des connaissances peut se restreindre en trois phases (cf. figure 2), de la manière suivante :

> « *L'**étape sémantique** concerne tout ce qui a trait au sens que l'on donne aux connaissances représentées*[15]*, ce qui passe par l'analyse et la structuration des concepts*

15. *. . . notre travail a consisté pour cette étape, à étudier de nombreuses monographies en Biologie et en Médecine et à participer à la réalisation de plusieurs bases de connaissances dans ces domaines. Cette analyse a conduit à différencier les entités signifiantes dans les descriptions des monographies et qui doivent être distinguées dans le formalisme (concept, descripteur, vraissemblance, ressemblance, individu, population, etc.)*

*correspondant au monde réel. L'**étape mathématique** concerne la définition abstraite du monde réel, dans laquelle une grande partie de la connaissance est explicitement représentée. Enfin l'**étape informatique** correspond à un codage des objets mathématiques, une résolution algorithmique du problème et sa mise en oeuvre par la programmation. Les aspects syntaxique et sémantique sont regroupés du fait du champ d'application précis choisi dans ce travail*[16]*, dans lequel les connaissances primitives sont déjà cernées.* »(Lebbe 1991)

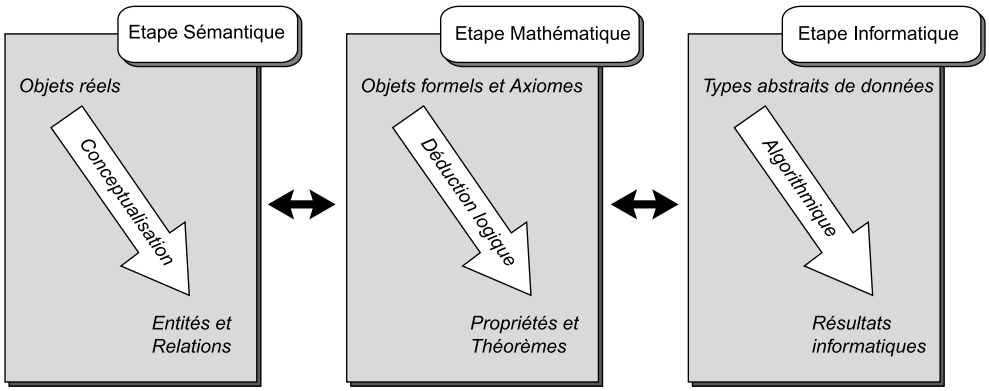


FIG. 2 – *Les étapes de la modélisation (Lebbe 1991)*

4 Positionnement et objectifs

Origine

Etant également concerné par la représentation des concepts et des descriptions en Biologie et en Médecine[17], nous adoptons un point de vue similaire concernant les trois phases de représentation des connaissances. Notre démarche diffère cependant de ces travaux sur deux aspects fondamentaux que nous développerons pendant cette thèse, les entités représentées et les moyens informatiques mis en oeuvre pour réaliser nos objectifs.

Nous nous situons également dans la continuité des travaux de Conruyt (1994) et de Renard et al. (1996) sur « la représentation et le traitement des connaissances descriptives pour l'amélioration de la robustesse des systèmes d'aide à la description, à la détermination et à la classification des objets biologiques » qui sont à l'origine de cette thèse.

Objectif principal

Dans la continuité de ces travaux, notre objectif principal consiste à élaborer des techniques permettant de construire des « ontologies d'un domaine » de la Systématique en général et de

16. *Représentation des concepts en biologie et en médecine* (Lebbe 1991)

17. Nous avons récemment conduit une application en *Coronaropathie* (étude des maladies coronariennes) en collaboration avec un groupe de cardialogues du CHD de Saint-Denis de la Réunion. Cette étude sort du cadre de cette thèse et ne sera pas exposée dans ce cadre.

la systématique des coraux en particulier [18]

4.1 Moyens mis en oeuvre

Les moyens mis en oeuvre principalement sont : une méthodologie de gestion itérative des connaissances, des techniques d'analyse (identification et classification) et d'extraction de nouvelles connaissances (synthèse de descriptions ou de concepts) ; ainsi que des outils informatiques adaptés à la démarche scientifique des systématiciens. Plus précisément, nous souhaitons offrir aux systématiciens les moyens de :

- Représenter les **taxons** d'un domaine particulier. Ce qui nécessite des outils de modélisation de concepts permettant de les organiser en hiérarchies.
- Décrire des **spécimens**. Avoir accès aux collections, et les décrire précisément à l'aide d'outils d'aide à la description.
- Elaborer un **thésaurus** consensuel du domaine, afin que les experts puissent s'appuyer sur une terminologie commune.
- Intégrer des **données d'ordre contextuel** : monographies, photographies, données biogéographiques, écologiques, bibliographiques, etc.. Outils de gestion d'information, bases de données, multimédia, etc..
- Appliquer les **méthodes d'analyse** qui ont été développées par les chercheurs en Analyse de Données et en Biologie [19], notamment pour l'identification et la classification d'objets biologiques. Il faut pour cela informatiser ces méthodes, qu'elles soient opérationnelles pour les données/connaissances représentées.
- Faire **évoluer** les données/connaissances. Outils de gestion de la dynamique des modèles et de mise à jour automatique des descriptions.
- Disposer de plusieurs **points de vue** possibles sur les données/connaissances. Les experts peuvent avoir des points de vue différents et des intérêts différents (morphologie, écologie, classification, etc.).
- **Transmettre** les connaissances. Par l'utilisation des « nouvelles technologies de l'information et de la communication » : multimédia (CD-ROM) et Web (langage Java, pages HTML, format XML, etc.).

4.2 Plan de l'exposé

Nous cherchons dans un premier temps (au chapitre 1) à définir clairement la terminologie et les concepts mis en oeuvre par les systématiciens dans leur travail. Dans un second temps, nous analysons différents modèles de représentation des connaissances élaborés par les informaticiens, en présentant leurs avantages et limites pour la problématique. Cette analyse détaillée nous conduit à mettre en évidence un ensemble de points clefs pour l'élaboration d'un modèle

18. Projet « Bases de connaissances sur les coraux des Mascareignes » à partir de la collection de plus de 3000 spécimens de G. Faure, en collaboration avec M. Pichon (EPHE) et M. guillaume (MNHN).

19. Depuis, les années années 60, les chercheurs se sont intéressés à la *Taxonomie Numérique* (Sneath & Sokal 1973, Diday 1971, Benzecri 1973)

conceptuel adapté à la représentations de descriptions complexes de spécimens biologiques et des concepts manipulés par la Systématique.

Nous proposons ensuite le modèle de représentation des connaissances appelé CoDesc (chap. 2), pour connaissances descriptives et classificatoires, élaboré pour les besoins spécifiques la Systématique. Nous montrerons de quelle manière ce modèle répond aux besoins des systématiciens à travers un ensemble d'exemples. Ce modèle conceptuel, opérationalisé au sein d'une architecture logiciel, est munie d'un ensemble de fonctions de manipulation des connaissances que nous exposons au chapitre 3 : recherche de l'APS, synthèse de descriptions généralisantes, filtrage, gestion de l'évolution, etc.

Dans la troisième partie de cette thèse nous développons un ensemble de méthodes d'*identification*, de comparaison et de classification d'objets biologiques prenant en compte la complexité des connaissances représentées. Pour l'identification, deux approches sont proposées. D'une part, une approche *monothétique* par arbres d'identification dynamiques (chap. 4), qui tient compte de la structure des descriptions représentées, des connaissances d'ordre stratégiques et qui est accessible "en-ligne". D'autre part, une méthode *polythétique* (chap. 5) fondée sur une « mesure de dissimilarité » originale, prenant en compte de manière très fine la richesse des descriptions. Cette mesure de comparaison est à la base d'un ensemble de méthodes de classifications : Classification Ascendante Hiérarchique, Nuées Dynamiques Distribuées et Classification Conceptuelle dont nous illustrons l'efficacité sur des données simples et structurées, issues de la « base de connaissances sur les coraux des Mascareignes ».

Dans la dernière partie de notre exposé, nous exposons différents aspects de la plate-forme à objets $\mathcal{IKBS}$[20], mettant en oeuvre une méthode *itérative* de gestion des connaissances, notre réponse méthodologique et informatique pour la conception des "Ontologies d'un domaine" de la Systématique.

20. *Iterative Knowledge Base System* ou *Système Itératif de Gestion de Bases de Connaissances.*

Première partie

Représentation des Connaissances en Sciences de la vie

Chapitre 1

Représenter des connaissances en Sciences de la vie

Sommaire

Quiconque s'est intéressé aux productions de la Nature, dont les êtres vivants sont les représentants les plus évidents, a perçu que, sous une apparence de diversité et de complexité extrême, se cachait en fait une sorte de plan d'ensemble, une régularité, une logique, un déterminime. Les naturalistes sont arrivés à la notion de « système de la nature », d'un ordre global dans lequel les différents individus se trouvent virtuellement regroupés en « classes », et ceci à différents niveaux ou « catégories » (Conruyt 1994).

1.1 Introduction

Nous proposons dans ce chapitre d'étudier précisément la terminologie et les différents concepts mis en oeuvre dans cette thèse, selon les points de vue des différentes communautés concernées par l'informatique appliquée à la Systématique.

Notre objectif principal est la mise au point d'un système à base de connaissances pour représenter, identifier et classifier les objets biologiques. Nous souhaitons prendre en compte la complexité des observations, qui s'exprime par la diversité et la variabilité des caractères descriptifs observés, en s'accommodant autant que possible des données manquantes, variables et imprécises, si fréquentes dans les domaines biologiques.

De cet objectif découle la révision apportée des concepts fondamentaux intervenant dans la classification des êtres vivants, au travers du point de vue des mathématiciens et des philosophes par rapport à celui des systématiciens. Nous montrerons en particulier dans quelle mesure l'acception des différents concepts peut diverger et se rejoindre [21].

Dans la seconde partie de ce chapitre, nous développerons les principaux modèles de représentation des connaissances élaborés par les différentes communautés de l'Informatique, en mettant en avant ceux qui peuvent être adaptés aux besoins des systématiciens. Nous discuterons à mesure des avantages et des limites de chacuns, et montrerons la nécessité d'introduire un modèle plus adapté à l'informatisation des données et des connaissances des systématiciens.

21. Notons que cette étude est la poursuite des travaux de Noël Conruyt (1994) sur l'« amélioration de la Robustesse des Systèmes d'Aide à la description, à la Classification et à la Détermination des Objets biologiques » qui sera poursuivit par Jacques Le Renard pendant sa thèse.

1.2 Extension et compréhension

1.2.1 L'extension

Deux points de vue de l'**extension** sont possibles selon le sujet d'étude et l'observateur.

Point de vue des philosophes et des mathématiciens

Les philosophes et les mathématiciens s'intéressent aux produits de l'activité humaine, c'est pourquoi l'extension est une notion dépendant de la **compréhension** : nous parlons de l'extension d'un concept par rapport à sa compréhension. Le sujet d'étude est la compréhension (ou intension) à partir de laquelle une extension est recherchée. Pour ces observateurs, l'extension est la sphère plus ou moins grande des êtres ou des espèces auxquels s'applique une condition, exprimée par un ou plusieurs attributs. La pensée organise spontanément les choses en classes les plus étendues, en éliminant de plus en plus de caractères. Aussi au plus l'extension croît, au plus la compréhension diminue.

Par exemple, tant que l'on ne connaissait pas de cygnes noirs, le concept cygne comportait dans sa compréhension l'attribut nécessaire blanc. Son extension comportait tous les cygnes connus (qui étaient tous blancs). Après la découverte de cygnes noirs, le concept cygne a perdu en compréhension l'attribut blanc (qui n'était plus nécessaire désormais) et a gagné en extension les nouveaux cygnes découverts.

L'extension peut être qualifiée de *psychique* ou *abstraite* car elle dépend d'une **définition** préalable des classes dans un univers de description donné (PClass et PConcepts (Sutcliffe 1993)). Dans ce contexte, il peut arriver que l'extension d'un concept soit vide : le concept de licorne par exemple (Sowa 1984).

En résumé, la classe traduit l'extension d'un concept, elle n'existe que lorsqu'elle a été explicitée : elle constitue l'ensemble des individus qui satisfont à la condition exprimée par son concept dans un univers de description donné.

Point du vue des biologistes et des naturalistes

Ces personnes s'intéressent plus aux produits de la nature, la compréhension n'a d'intérêt que si elle traduit une extension concrète. Ainsi l'extension peut être une notion indépendante de la compréhension. Le sujet d'étude est l'extension et l'on considère que les classes préexistent avant même de recevoir une définition. Par exemple, un chien qui passe dans la rue existe indépendamment de sa définition. Chaque classe correspond à une certaine extension (ou couverture) concrète et naturelle dont on veut tirer un enseignement (une compréhension des classes naturelles).

Dans un premier temps, on se contente donc de décrire l'extension ou le contenu (les individus) de la classe sous forme de **descriptions**. Le fait de décrire est déjà en lui-même un enseignement pour le descripteur (celui qui décrit). Il est amené à interpréter des observations multiples et hétérogènes afin de produire des généralisations "de bas niveau" (en ne mesurant que certaines propriétés et en en ignorant d'autres) supposées exactes et dignes de confiance.

Les descriptions doivent tenir compte de la diversité biologique exprimée par la couverture

de la classe[22].

Les biologistes s'efforceront donc de traduire cette diversité dans les descriptions, afin de recueillir toute la richesse et la diversité des individus du domaine biologique bien délimité. En effet, chaque individu décrit est un élément représentatif de la classe et a pour extension lui-même : notre approche privilégie ainsi la multiplication des descriptions d'individus appartenant à une même classe (avec des valeurs comprises dans un intervalle de doute ou d'imprécision) plutôt qu'une seule description de "concept" dont l'extension est l'ensemble des individus qui vérifient l'intervalle de variation des valeurs de la description. Cette deuxième approche est celle adoptée par (Vignes 1991). La formalisation sous forme d'objets symboliques (Diday 1987) met aussi en lumière cette nuance.

Ainsi comprises, les descriptions forment une base de travail exhaustive pour le traitement et constituent déjà un résultat important pour la transmission du savoir humain. Dans un deuxième temps, le dscripteur cherche à mieux comprendre ses descriptions individuelles.

1.2.2 La compréhension

La compréhension ou l'intension est l'ensemble des caractères ou propriétés contenus dans un concept et qui permettent de le définir (Arnault & Nicole 1662).

Ainsi vertébré a comme compréhension : animal qui a des vertèbres et comme extension Mammifères, Oiseaux, Batraciens, Reptiles, Poissons. Nous pouvons remarquer qu'un concept s'étend à d'autant plus d'êtres qu'il réunit moins de caractères, comme le montre la figure 1.1.

Ainsi la compréhension et l'extension sont en raison inverse l'une de l'autre. Animal a une extension plus stricte et une intension plus forte que Vivant, Vertébré a plus de compréhension qu'Animal et Mammifère plus que Vertébré.

D'après cette définition de la compréhension, l'intension est la partie signifiante du concept. Elle énonce certaines propriétés (supposées vraies) permettant de valider des connaissances du domaine. Elle exprime les conditions nécessaires et/ou suffisantes[23] d'appartenance d'un individu

22. Chaque objet de l'extension possède un statut avec différentes modalités que le descripteur peut être amené à envisager :
ces informations sont **évolutives** : elles changent au fur et à mesure que l'univers dans lequel elles sont utilisées se modifie, ce qui entraine des problèmes de cohérence.
certaines ou incertaines : il peut résider ou non un doute quant à la vérité des informations. Ce doute peut être dû à un manque de confiance dans la source de l'information ou au fait que celle-ci est difficilement accessible à la vérification.
valides ou périmées : elles n'ont pas toujours de valeurs universelles et peuvent être remises en question dans l'avenir.
typiques ou exceptionnelles : chaque objet, qu'il soit considéré comme central ou marginal, porte sa propre originalité et fait ainsi partie intégrante de la couverture de la classe. A ce titre, les cas exceptionnels ont autant d'importance que les cas typiques en biologie, c'est pourquoi les biologistes affirment que dans la nature, l'exception est la règle. Il ne s'agit donc pas de les supprimer !
complètes ou incomplètes : la connaissance disponible sur un objet est généralement incomplète parce qu'elle est implicite et donc généralement oubliée dans la représentation, ou encore parce qu'elle n'est pas encore connue ou qu'elle est difficile à transmettre.
significatives ou fictives : des informations ont un sens pour expliciter des règles de connaissances alors que d'autres ne sont utilisées que pour structurer le domaine de description.

23. Une définition n'est pas forcément nécessaire et suffisante du fait que des personnes différentes ont rarement

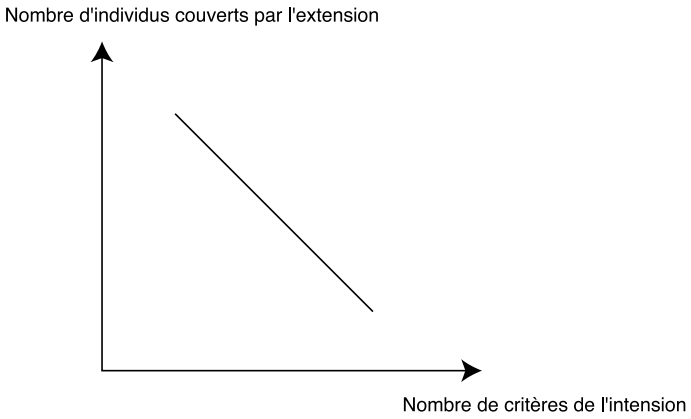


FIG. 1.1 – *Rapport entre l'extension et l'intension.*

au concept.

Néanmoins, la question de savoir si l'intension prime sur l'extension est un problème philosophique qui dépend de l'observateur et de ses préoccupations. En effet, l'intension précède-t-elle l'extension dans la vision que possède l'utilisateur du domaine étudié ? Il semble naturel que la réponse soit oui pour un psychologue et un mathématicien : la définition ne peut être faite que par l'homme ! Ce à quoi le naturaliste objectera en posant la question suivante : est- ce que les animaux qui existaient au secondaire et que l'on a appelés dinosaures par la suite faisaient partie d'une classe ? Est-ce qu'ils existaient avant que le concept n'apparaisse ? Il semble aussi que oui ! Nous sommes ainsi en présence d'une dualité de point de vue résumée par la figure 1.2.

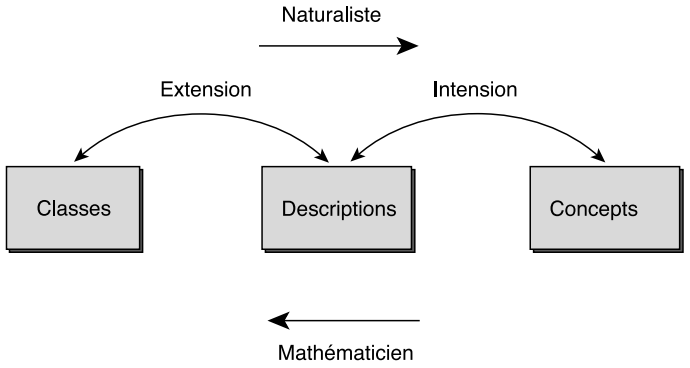


FIG. 1.2 – *Mathématiciens et Naturalistes, deux points de vue différents des concepts.*

Ce schéma un peu caricatural demande un approfondissement dans l'étude des raisonnements différents qu'employent ces deux catégories de personnes :

Le mathématicien a l'habitude d'utiliser un raisonnement démonstratif basé sur une valeur de vérité d'une propriété, exprimé par des règles rigoureuses et clarifiées par la logique formelle. Ce type de raisonnement est sûr, à l'abri des controverses et définitif. Inversement, le naturaliste émet des hypothèses qu'il justifie par un raisonnement plausible. Ce dernier est hasardeux, il peut être controversé et il est provisoire (Pólya 1958). Néanmoins, il est capable de conduire à des

la même compréhension d'un même phénomène naturel : voir plus loin les définitions des intensions minimales, strictes et généralisées des concepts au § 1.4.

connaissances essentiellement nouvelles sur le monde qui nous entoure. C'est pourquoi ces deux types de raisonnement ne sont pas contradictoires comme pourrait le laisser penser le schéma ci-dessus : ils se complètent.

Dans le raisonnement rigoureux, l'essentiel est de distinguer une preuve d'une présomption, une démonstration valable d'une tentative qui a échoué : c'est le *savoir démontrer* du mathématicien qui prouve la validité de ses concepts. Dans le raisonnement plausible, l'essentiel est de distinguer une présomption d'une autre, l'une plus raisonnable que l'autre : c'est le *savoir pressentir* du naturaliste qui suggère des classes fiables. Le mathématicien doit donc être capable de deviner une règle ou un théorème mathématique avant de le démontrer, de même que le naturaliste devrait être capable de prouver le bien fondé de ses règles de classification. Il est donc faux d'opposer la démarche d'un naturaliste à celle d'un mathématicien comme voudrait le laisser paraître notre monde contemporain assoiffé de démonstrations et de certitudes.

Dans cette thèse, en tant que cogniticien, nous nous plaçons d'abord du point de vue de la demande du naturaliste qui considère l'extension comme un sujet d'étude.

Démarche du naturaliste

Partant d'une classe (ensemble d'individus) dont le contenu (l'extension) est sa couverture, le naturaliste observe et crée le nom de cette classe puis... la définit (en intension) afin de créer le concept associé. Cette démarche constate d'abord la classe avant de procéder à une conceptualisation de ses individus.

Certains mathématiciens comme Euler (1707-1783) ou Laplace (1749-1827) prônent ce point de vue basé sur l'observation. Néanmoins, contrairement au naturaliste pour qui l'observation est le critère le plus élevé (la vérification effectuée dans de nombreux cas bien choisis est la seule méthode de confirmation d'une loi hypothétique dans les sciences naturelles), le mathématicien va plus loin dans son domaine en affirmant que si nombreuses que puissent être des vérifications expérimentales, elles ne suffisent pas à démontrer que la loi supposée est vraie. Cette bifurcation de point de vue tient donc à la nature du domaine étudié (réfutabilité des hypothèses) : la récurrence et la périodicité ne se rencontrent pas dans la nature !

Ensuite, nous nous plaçons du point de vue de l'informaticien dont l'approche est située entre la démonstration et l'observation et qui fournit une offre en matière de modèles formels et de technologie.

L'informaticien agit au niveau des descriptions : il donne la possibilité avec les outils qu'il développe de normaliser les observations des naturalistes, élevées au rang de descriptions comparables entre elles car utilisant le même schéma de représentation. C'est à partir de ces descriptions que nous allons bâtir des hypothèses "plausibles" par induction et que nous allons les vérifier grâce à l'identification de nouvelles observations. Par les outils que l'informaticien fournit, nous serons capables d'appliquer la méthode hypothético-déductive chère à Popper (Popper 1973, Popper 1978) :

« *La méthode de la Science est une méthode de conjectures audacieuses et de tentatives ingénieuses et sévères pour réfuter celles-ci* ».

1.3 La classe

C'est l'ensemble ou groupe d'individus (point de vue du naturaliste), possédant tous un ou plusieurs caractères communs et étant les seuls dans ce cas (point de vue du philosophe ou du mathématicien).

1.3.1 Point de vue des mathématiciens

Pour eux comme pour certains philosophes (suivant en cela la tradition d'Aristote), la classe dérive du concept : il s'agit d'un ensemble d'objets qui satisfont une condition prédéfinie nécessaire et suffisante (dans un univers de discours donné) et qui forme ainsi l'extension d'un concept (Sutcliffe 1993). Cette sorte de classe peut être nommée classe conceptuelle (Niquil 1993).

Il existe toutefois une partie des mathématiques qui considère les objets sous leur aspect extensif et que l'on peut qualifier d'expérimentale car basée sur l'induction (Euler 1747).

Néanmoins, la partie la plus importante des mathématiques "modernes" (la théorie des ensembles, la logique formelle, les prédicats) s'intéresse plutôt à leur aspect compréhensif (Frege 1893) et à la déduction. Le but de cette dernière approche est de calculer l'extension du concept $\mathcal{C}$ en définissant une application $a_{\mathcal{C}}$ de l'ensemble des individus observés $\Omega \rightarrow [vrai,\ faux]$ qui à chaque individu ω de Ω fait correspondre son appartenance au concept $\mathcal{C}$ ou non.

$$\begin{aligned} a_c :\ & \Omega \rightarrow [0,\ 1] \\ & \omega \mapsto 1 \text{ si } \omega \in \mathcal{C}, \\ & \omega \mapsto 0 \text{ sinon} \end{aligned}$$

Les individus sont ainsi baptisés **instances** du concept s'ils appartiennent au concept [24]. Comme nous l'avons déjà expliqué plus haut (§ 1.2.1), l'extension dépend de la compréhension pour certains alors qu'elle est le point de départ pour découvrir une intension pour d'autres.

Donc, pour le mathématicien, la classe n'existe que si elle est *explicitée en intension* (dans le monde des idées) selon un certain point de vue et correspond à un concept. Elle peut être qualifiée d'**abstraite**. Prenons garde néanmoins au terme d'existence : une définition n'entraine pas l'existence de la chose définie, les objets mathématiques étant donnés au départ par postulat (les fonctions, les nombres, le cercle, etc.) (Bourbaki 1974).

1.3.2 Point de vue des systématiciens

Définition de la classe :

(Histoire naturelle) : "Bien que, comme tous les groupes plus vastes que l'espèce, la classe soit un concept en partie abstrait (un niveau taxonomique), on donne à de nombreuses classes

24. Une partie plus récente des mathématiques s'intéresse au degré d'appartenance "flou" des individus à des concepts (Zadeh 1965) : un élément appartient plus ou moins à un ensemble. En ce qui concerne les spécimens, le naturaliste n'est pas habitué à jongler avec l'incertitude et l'imprécision pour attribuer un individu à un concept, il finit par trancher. Cette caractéristique étant naturelle en biologie, nous n'avons pas étudié plus avant la théorie des possibilités (Dubois & Prade 1987) pour l'appliquer dans la représentation des connaissances du domaine.

une définition tout à fait précise, correspondant au fait que les êtres de cette classe possèdent tous un certain caractère et sont seuls à le posséder. Les Insectes ont tous un thorax formé de trois anneaux et portant trois paires de pattes articulées ; les Oiseaux ont tous des plumes ; les Monocotylédones ont toutes un embryon à un seul cotylédon ; les Céphalopodes ont tous des tentacules, etc.." (Larousse)

Remarque : Les systématiciens employent le mot Classe (ou *Classis*) avec une majuscule pour désigner l'une des catégories de la systématique comprise entre les Ordres (*Ordo*) et les Embranchements (*Phylum*). Quoi que traitant ici de systématique, nous n'emploierons jamais le mot classe dans ce sens strict.

Nous l'employerons plutôt comme synonyme de groupe (ou taxon) à un certain niveau hiérarchique :

(Histoire naturelle) : "Subdivision usitée en classification zoologique ou botanique et dont on ne peut pas ou on ne veut pas préciser la valeur hiérarchique : Classe, Ordre, Genre, Embranchement, etc.." (Larousse)

La première définition précédente de la classe, si on l'étend aux différents taxons de la classification linnéenne, donne comme exemples des caractères propres ce qui a été appelé "caractères dominateurs", entièrement caractéristiques d'une classe. Dans les faits, il est rare qu'une classe puisse être ainsi caractérisée par un caractère unique. La diversité biologique que l'on constate dans la nature fait que la définition d'une classe regroupe généralement la conjonction de plusieurs caractères. La définition semble de plus considérer le terme de concept comme synonyme de classe, ce qui ne correspond pas à notre analyse.

Pour ces raisons, nous considérons que cette définition ne correspond pas toujours à la réalité des choses. Par le terme équivalent de concept, elles apparaissent comme des intensions, des preuves sûres, démontrables au sens mathématique et pas du tout comme des hypothèses plausibles, vraisemblables et à vérifier par de nouveaux faits.

Chaque classe naturelle peut être :

1. **Nommée**. On peut s'y référer sans ambiguïté par son nom ; c'est un principe, magistralement arrêté par Linné (1735), que la découverte de toute nouvelle classe doit être accompagnée par son auteur de la fixation d'un nom ; cette dénomination doit respecter des règles de nomenclature bien définies (binôme spécifique, loi de priorité, etc.), en particulier pour s'assurer de son unicité.
2. **Définie par son contenu**. La classe peut être définie concrètement par son **contenu** (sa population), représentée par exemple par l'énumération des individus connus qui composent son effectif. De façon plus pragmatique, on se contente d'un échantillon "représentatif" de la population, qui doit illustrer au mieux la variabilité naturellement présente. Il se peut qu'il y ait parmi les **descriptions d'individus**, à la fois des descriptions d'un seul spécimen (un individu réel) et des descriptions synthétiques de plusieurs spécimens (individu virtuel).
3. **Caractérisée par des traits propres**. La classe peut être caractérisée, de façon aussi discriminante que possible, par un ensemble de caractères propres à la distinguer des autres classes, dont l'énoncé constitue sa **diagnose**.
4. **Typifiée**. La classe possède un spécimen type, que son auteur a choisi pour la représenter de façon unique et définitive. Ce type n'est pas nécessairement un prototype ou un individu

"moyen". Une classe en tant que concept biologique n'existe que si un type lui a été associé. L'*Homo sapiens* est la seule espèce qui ne possède pas de type.

5. **Conceptualisée**. La classe peut être envisagée comme un **concept**, une fois qu'elle a été définie, chacun de ses individus apparaissant à la fois comme un représentant du concept et comme un élément objectif (faisant partie de la couverture) ou subjectif (conforme à la définition) de la classe.

1.4 Les concepts

Les concepts sont considérés du point de vue de la compréhension qui désigne l'ensemble des caractères exprimés par le mot, et du point de vue de l'extension, qui désigne l'ensemble des individus auxquels le mot s'applique.

Un concept est une abstraction intellectuelle de parties du monde. C'est une idée *abstraite* (obtenue en se bornant à considérer certains caractères des objets, à l'exclusion d'autres caractères pourtant perceptibles) et *générale* (étendant les caractères ainsi considérés à tous les objets qui les possèdent). Tout concept se caractérise par sa compréhension (ensemble des caractères considérés dans les objets) et par son extension (ensemble des objets auxquels il peut s'appliquer). Compréhension et extension forment donc l'aspect logique du concept une fois élaboré (LConcept). Abstraction et généralisation sont les deux opérations psychologiques par lesquelles il s'élabore (PConcept) (Sutcliffe 1993).

Chez Aristote, on trouve la notion de *logoi* pour le concept avec deux points de vues : l'un considère les sujets que regroupe la classe correspondante au concept et l'autre est le prédicat qui est la condition d'appartenance d'un sujet à la classe du concept. Il y a trois façons (logoi) de se référer à un concept :

1. **Par son contenu** (l'être). L'être est le référent ou l'extension du concept. C'est l'ensemble des instances du concept (les choses existantes auxquelles le concept s'applique).
2. **Par sa définition** (l'essence). L'essence est la condition d'appartenance à la classe. On donne un prédicat ou définition (une condition) ce qui crée le concept en intension (le nom n'est pas forcément présent).
3. **Par son nom** (terme univoque qui abrège la définition). Le nom du concept est un abrégé ultime de la définition. Il peut faire intervenir la propriété la plus caractéristique pour le résumer (par exemple, la rouille du blé [25], un réfrigérateur, etc.). Néanmoins, le nom est avant tout une commodité, un code de reconnaissance, qui est difficilement utilisable si l'on fait abstraction de sa définition complète (ambiguité). En sciences naturelles, le nom est donné en latin pour lui conférer un caractère universel.

1.4.1 Du point de vue naturaliste

Dans notre approche des concepts, nous affirmons leur existence dès lors que nous fixons :

– **Une classe**. Pour les biologistes, la classe est une vérité ; elle a une existence naturelle avant même d'être définie en tant que concept.

25. Maladie fongique caractérisée par des tâches de couleur rouille.

- **Une définition** associée à la classe. Pour le concept, ce n'est pas le nom qui est important mais bien l'intension qui lui est attribuée (sa définition). Un concept est délimité par la définition de la classe correspondante.
- **Un univers de discours** (un contexte). La définition de la classe dépend du contexte : il peut exister en effet différents concepts associés à une même classe. Par exemple, le concept de "grand homme" dépend de l'univers de discours pour sa définition. S'agit-il du sens donné à la taille d'un individu ou bien celui de sa réputation ? Napoléon et Charles de Gaulle ne seraient pas classifiés de la même manière selon le contexte choisi ! De même, la classe des tomates ne correspond pas à la même définition chez un botaniste et chez un cuisinier : c'est un fruit pour le premier et un légume pour le second.
- **Une capacité d'abstraction intellectuelle** plus ou moins élaborée. La définition de la classe dépend du niveau de perception. Par exemple, le concept de dinosaures pour un paléontologiste correspond à un stade d'évolution dans la lignée des reptiles alors que le concept de dinosaures pour un enfant peut correspondre à celui d'un monstre sympathique, personnage de dessin animé.

Pour un univers de discours donné et un certain niveau de perception, un concept associé à la classe peut être déterminé. Un concept est déterminé lorsque l'on explicite les caractères compréhensifs du concept (Petit-Robert, 1994).

A chaque concept, on peut associer plusieurs niveaux de définition de la classe considérée :

1. L'**intension généralisée** donne des conditions *nécessaires* d'appartenance à la classe. Ces conditions forment une généralisation [26] de la classe et la définition obtenue ne se trouve donc que **partiellement observée**. Tous les individus qui y appartiennent satisfont à cette définition. Néanmoins, il peut y avoir des individus qui n'appartiennent pas à la classe mais qui sont conformes à la définition.
2. L'**intension stricte** exprime des conditions *nécessaires* et *suffisantes* d'appartenance à la classe : tout individu qui satisfait à l'intension stricte de la classe en fait partie. Inversement, tout individu qui appartient à la classe satisfait à son intension stricte. Chacune des conditions exprime une régularité **intra-classe**. L'intension stricte est une intension **observée**, elle est issue d'une simple reformulation [27] de la disjonction des descriptions réelles de la classe (par factorisation, par la prise en compte de connaissances de fond, etc.). Elle est *absolue* car elle ne fait pas intervenir les définitions des autres classes.
3. L'**intension réduite** ou *diagnose stricte* donne le plus petit jeu de conditions nécessaires et suffisantes d'appartenance à la classe. A partir de l'intension stricte, nous pouvons dériver une intension réduite ou **diagnose stricte** qui donne le plus petit jeu de conditions nécessaires et suffisantes d'appartenance à la classe. Chacune de ces conditions correspond à une différence **inter-classe**. Il faut remarquer que cette caractérisation succinte est relative aux autres définitions de classes que l'on veut comparer pour être en mesure d'évaluer leurs différences : elle n'est pas absolue du fait qu'elle doit être modifiée à chaque fois qu'une

26. La généralisation peut être définie comme un ajout d'observable à de l'observé. En effet, le résultat de la généralisation englobe des situations intermédiaires **observables**, non effectivement observées.

27. Une reformulation est une formule comprimée de l'intension par réécriture, elle est plus dense, mais elle contient la même information (iso-intension) et le même contenu au niveau de l'extension (iso-extension).

nouvelle classe est prise en considération. Il s'agit en effet d'une "connaissance croisée" (différentielle) dont on a retiré tout ce qui est commun avec les autres définitions de classe. La diagnose, issue d'une intension stricte, est une diagnose **observée**.

4. L'**intension modale** ou **typique** donne des conditions *suffisantes* d'appartenance à la classe. Tout individu (typique) répondant à cette définition "caractéristique" de la classe en fait partie[28]. Il peut y avoir néanmoins dans la classe des individus atypiques s'écartant de la définition de cette classe.
5. La **diagnose modale** ou **typique** est obtenue par réduction de l'intension typique par rapport aux autres classes. La plupart des "diagnoses" utilisées par les biologistes (surtout les botanistes) sont modales (elles évacuent les exceptions pour gagner en signification) ; elles comportent souvent une part plus ou moins importante de généralisation pour en faciliter la compréhension par le profane.

1.4.2 Du point de vue mathématique

Pour formaliser ce que nous venons de dire, nous donnons les définitions suivantes :

Soient $\Omega = \{\omega_1, \ldots, \omega_n\}$, l'ensemble des spécimens ou individus observés,

$\Pi \supset \Omega$, l'ensemble de tous les individus observables,

$P(\Omega)$, l'ensemble des parties de la population observée Ω.

Soit F, une fonction de représentation de $\Pi \to O$, O désignant l'espace d'observation, qui à chaque individu observable ω de Π fait correspondre sa description potentielle $y(\omega) = \delta \in O$:

$$\begin{aligned} F :&\Pi \to O \\ &\omega \mapsto F(\omega) \end{aligned}$$

Soit y, une fonction de représentation de $\Omega \to \Delta$, Δ désignant l'espace de description des individus observables ($\Delta = F(\Pi) \supset O$), qui à chaque individu observé ω de Ω fait correspondre sa description $d = y(\omega) \in \Delta$:

$$\begin{aligned} y :&\Omega \to \Delta \\ &\omega \mapsto y(\omega) \end{aligned}$$

Soit une classe observée $C \in P(\Omega)$. Pour chacune, on peut associer une définition $D = y(C), D \in P(\Delta)$. En notant b_D la fonction d'appartenance à la classe D :

$$\begin{aligned} b_D :&\Delta \to [0,1] \\ &d \mapsto 1 \text{ si } d \in D, \\ &d \mapsto 0 \text{ sinon} \end{aligned}$$

28. le modèle de classe ou modèle à objets est fondé sur ce principe.

D représente la somme (ou disjonction) des descriptions observées de chaque individu de la classe : $D = \Sigma d.\ d \in D$ est aussi appelé un exemple de la classe D, un contre-exemple est donc un élément de $\delta \setminus D$.

On obtient ainsi le schéma de la figure 1.3 présenté dans (Diday 1993).

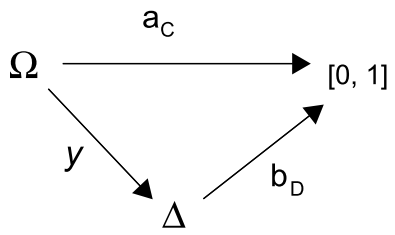


FIG. 1.3 – *Le triangle des fonctions entre individus et leurs descriptions.*

avec la propriété : $\forall \omega \in \Omega,\ a_C(\omega) = b_D(y(\omega)) = b_D \circ y(\omega)$

Le schéma de la figure 1.4 synthétise le formalisme :

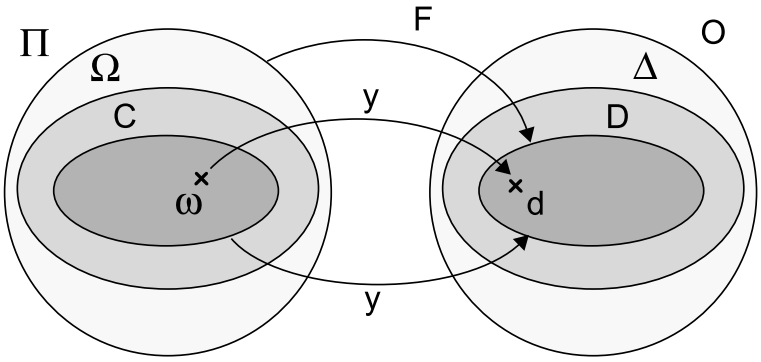


FIG. 1.4 – *Schéma du formalisme de modélisation des données.*

1.5 Classement et classification

1.5.1 Classer et le classement

Classer consiste dans un premier sens à regrouper des individus ou des objets afin de former des classes. Chacune d'elles se voit attribuer un nom (une étiquette). Classer est une action en deux étapes : à partir d'un tas d'individus, on effectue un tri en répartissant les objets selon leurs ressemblances et différences (on établit une partition des objets), puis on étiquette chaque groupe ainsi formé par un nom de code. Il existe un second sens au verbe classer qui est celui de déterminer : assigner la classe à laquelle appartient une chose, un individu. Nous préférerons employer le terme déterminer pour la seconde acception.

Le classement, selon les deux sens attribués au verbe classer, permet dans un sens de constituer des **regroupements nommés d'objets** *a priori* afin de former des **classes concrètes** (définies en extension par les objets qu'elles possèdent) et, dans l'autre, à retrouver le nom d'un nouvel individu *a posteriori* par rapport aux classes déjà formées. Le classement *a priori* est une

démarche exploratoire sur un ensemble d'objets dont on ne perçoit aucune définition en l'état (ou dont la définition n'a pas d'intérêt immédiat). Une personne naïve dans un domaine est capable d'effectuer ce classement.

Le classement *a posteriori* permet l'identification des objets entre eux de manière globale en partant de la classe. Il s'agit d'un processus de comparaison directe des objets entre eux qui ne nécessite pas forcément l'usage de descriptions de ces objets, et moins encore d'une quelconque définition de ces individus.

1.5.2 Classifier et la classification

Classifier, c'est conceptualiser des classes, c'est-à-dire les créer par classement, puis les définir, et les nommer éventuellement. Classifier est une des fonctions essentielles de l'intelligence humaine : elle repose sur un plus grand niveau d'expertise que le classement. Cette notion est souvent confondue avec déterminer ou identifier en intelligence artificielle où l'on parle de classifier des observations lorsqu'il s'agit de trouver le nom de la classe auxquels elles se rapportent. En effet, pour certains statisticiens et mathématiciens, la classification veut dire la même chose que le classement *a posteriori*.

La classification, prise dans le sens des systématiciens ("classification des êtres vivants") est la faculté de former un classement (en partitionnant), puis par un regroupement donné d'individus, de formuler une définition de ce groupe. Le résultat s'appelle une classification. Il s'agit de représenter les caractéristiques de chaque classe : on établit ainsi des **classes abstraites définies en intension par des concepts** (et non plus par des objets). De plus, la classification cherche à hiérarchiser les classes selon leur degré de généralité afin de former différents niveaux taxonomiques.

La classification en Analyse des Données n'est pas nécessairement conceptuelle : aucune définition des classes n'est extraite à partir des données.

Dans toute science, il est nécessaire de classifier les phénomènes et les objets que l'on veut étudier et ceci est particulièrement vrai dans les sciences qui étudient les êtres vivants. Une classification vraiment scientifique des végétaux et des animaux doit être naturelle et non artificielle, c'est-à-dire fondée non sur des caractères arbitrairement choisis pour une raison de commodité ou d'utilité quelconque, mais sur les caractères les plus importants du point de vue de la structure anatomique des êtres et de leurs grandes fonctions physiologiques. Les classifications de l'histoire naturelle se proposent d'indiquer le degré de ressemblance et de différence réelle, et non pas apparente et superficielle, de chaque être avec tous les autres. Certains auteurs affirment (d'autres nient) que ces ressemblances sont l'expression d'une *parenté* généalogique entre les espèces et qu'une bonne classification doit tendre à mettre en évidence la phylogénie des groupes, c'est-à-dire la suite des formes que l'évolution leur a fait parcourir.

La classification est la partie noble du classement. Elle consiste à ranger dans un même groupe (une classe au sens du biologiste) et à désigner du même nom des faits, des objets ou des êtres qui possèdent en commun certains caractères. Elle suppose l'analyse, la comparaison, mais plus encore la faculté de faire abstraction des différences individuelles. La formation d'une idée générale est un acte de classification. Cette formation s'appuie sur la capacité à décrire les individus, de les classer et de les nommer avec une étiquette, puis de les définir par une intension :

cette capacité est le propre de l'expert du domaine, comme l'illustre la figure 1.5.

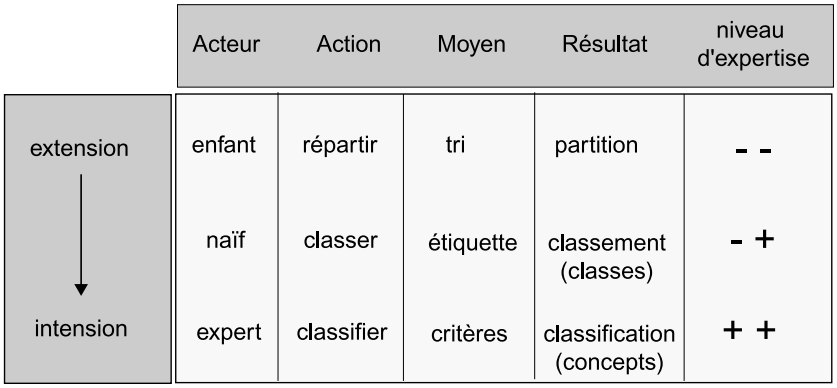


FIG. 1.5 – *Schéma de comparaison des termes employés en systématique.*

La classification s'accompagne de la **caractérisation** des classes (obtenues de manière expérimentale ou artificielle) : elle recherche les critères représentatifs (ou caractéristiques) de la classe (par confirmation des ressemblances intra- classe) et les critères de différenciation (ou de discrimination) des classes (par élimination des différences inter-classe). Elle permet d'expliciter les classes à partir des descriptions d'individus (explicitant elles-mêmes les individus des classes). La classification procède par **généralisation inductive** des descriptions, elle est une démarche **synthétique**. Cette synthèse permet de créer des connaissances nouvelles que l'opérateur espère meilleures pour comprendre son domaine.

Deux sortes de classification "artificielle" sont évoquées parmi les méthodes d'apprentissage des descriptions qui nous intéressent :

Classification "non-supervisée"

La première sorte procède à partir de descriptions d'un échantillon du domaine étudié sans connaissance préalable du nom associé à chacune d'elles. Ces descriptions sont appelées **observations** en apprentissage automatique car elles ne possèdent pas d'identification associée (on parle aussi d'apprentissage sans professeur). Le but consiste ici à découvrir les classes et/ou les concepts cachés dans les observations.

Ce type de démarche classificatoire, classique en Analyse des Données (méthodes factorielles (Benzecri 1973), nuées dynamiques (Diday 1971), et en taxonomie numérique (Sneath & Sokal 1973), est aussi appelé **catégorisation** (Napoli 1992) ou classification conceptuelle (Fisher & Langley 1985). Il procède par agrégation des observations selon leurs **ressemblances** avec certaines mesures de similarité puis **caractérisation** en interprétant les classes obtenues par un ensemble de caractères propres permettant de définir les concepts associés.

Le regroupement conceptuel est le même type de classification dans le secteur de l'intelligence artificielle et qui tient compte en plus de connaissances sur le domaine (Stepp & Michalski 1986).

Classification "supervisée"

La seconde sorte de classification opère à partir d'**exemples** ou de *cas* qui sont des descriptions d'individus observés auxquelles l'expert a attribué un nom (une étiquette ou bien encore une identification associée après classement) : là, on connaît le concept à apprendre (la maladie, l'espèce, etc.). Ce type de classification avec professeur (ou supervisé) est encore divisé en deux sortes :

1. Le premier, qualifié de "descendant", est appelé **discrimination** à partir d'exemples et procède par **segmentation** des cas selon leurs différences, en fonction de certains critères : fonction coût (Hunt et al. 1966), gain d'information (Quinlan 1979), réduction d'impureté (Breiman et al. 1984), etc.
2. Le second utilise une stratégie ascendante guidée par les données dont l'*algorithme de l'étoile* avec les systèmes AQ (Michalski 1983) est le représentant le plus typique.

1.6 Détermination et identification

La détermination

Comme pour la classification, la **détermination** peut avoir une double signification opposée : d'une part, on parle de la détermination d'un concept lorsqu'il s'agit de le définir ou de le caractériser ("déterminer un concept" est alors équivalent à "classifier"). D'autre part et de façon plus courante, le mot est employé pour désigner l'action inverse de la classification : c'est une démarche qui permet de déduire l'appartenance d'un individu à une classe en utilisant sa définition en intension : cette démarche est **analytique**. Dans ce sens, il n'y a pas de détermination possible sans classification préalable. Nous souhaitons bien distinguer les deux aspects inductif et déductif de la démarche scientifique dans cette thèse. C'est pourquoi nous emploierons la détermination dans le sens déductif opposé à la classification inductive.

De plus, la détermination ne doit pas être confondue avec l'identification : déterminer permet de trouver le nom de la classe ou le concept associé à la nouvelle observation. Le procédé permettant de passer d'un indéterminé (individu ou spécimen que l'on peut observer et/ou décrire) à un déterminé (indéterminé affecté à une classe d'identification) est nommé détermination.

L'identification

L'**identification** s'applique plus au domaine de l'extension contrairement à la détermination qui concerne le domaine de l'intension : dans le langage courant, identifier est employé plus souvent pour trouver le nom d'un individu (la plupart du temps un humain), ou un code qui permet de se référer à l'identité de quelque chose. On dit plutôt "identifier un individu" pour dire que l'on a trouvé son identité, plutôt que "déterminer un individu". Inversement, on parlera de "déterminer la classe d'un individu" lorsque l'on utilisera une définition de son concept.

Alors que la classification est affaire de spécialistes, il est fréquent que la détermination soit conduite par un "béotien" en la matière, comme ce douanier qui doit déterminer s'il a devant lui

un animal protégé ou non par la convention de Washington, ou lors d'un recensement écologique où il est nécessaire de distinguer (et de désigner) les différentes espèces en présence.

Toute détermination se fait par référence à un corpus de connaissances préexistant, qu'il soit organisé (clef de détermination, système expert, etc.) ou non (livres, connaissance résultant d'un apprentissage plus ou moins empirique).

Il faut aussi remarquer qu'une détermination ne conduit pas toujours à un résultat certain, du fait d'inexactitudes ou d'imprécisions soit dans les connaissances de référence soit dans la possibilité ou la capacité d'observer correctement l'individu à déterminer. De plus, la précision attendue pour une détermination doit être adaptée à l'utilisation prévue du résultat ; les applications dans le domaine scientifique sont bien sûr les plus exigeantes.

Selon les cas, plusieurs situations de détermination peuvent se rencontrer, isolément ou en concours.

1.6.1 Détermination par comparaison directe

Ce premier mode de détermination exige la disposition d'une collection de référence (herbier, jardin botanique par exemple) ou d'un substitut (flore où les différentes espèces sont figurées). Il suffit (non sans mal néanmoins !) de comparer visuellement l'indéterminé avec chacun des référents disponibles, afin de sélectionner celui qui correspond le mieux ; le nom de ce référent est alors adopté comme l'identification recherchée.

Du fait que cette méthode n'astreint pas à décrire, la qualité du résultat est étroitement dépendante des dons d'observation du déterminateur. Tout tient en effet en sa capacité de juger de "l'identité" entre deux individus, qui ne sont pourtant jamais semblables s'agissant de créatures de la nature. Comme aucun contrôle n'est possible, puisqu'aucune connaissance n'est *a priori* pré-requise, elle peut conduire à des erreurs quand l'oeil n'est pas suffisamment exercé.

Elle constitue par contre l'ultime confirmation pour le spécialiste, pour lequel la comparaison visuelle directe avec le type demeure l'épreuve de vérité irremplaçable. Le type est l'unique spécimen désigné comme le référent absolu de chaque classe lors de la création de celle-ci ; il n'existe pas de classe dépourvue de type, sauf celle de l'Espèce *Homo sapiens* peut-être pour des raisons éthiques.

1.6.2 Détermination par comparaison avec des descriptions

Ce deuxième mode nécessite d'abstraire le spécimen indéterminé, en en faisant la description plus ou moins complète. La seule observation n'est plus suffisante. En effet, la comparaison va se faire non plus avec des référents concrets, mais avec des descriptions jouant le rôle de référents abstraits. Chaque classe naturelle est pourvue, outre son type, d'une description ou d'une diagnose (description différentielle) ; chaque flore ou chaque faune constitue ainsi un recueil de descriptions, équivalent en quelque sorte de la collection de référence utilisée pour la comparaison concrète.

On procède par élimination progressive. Pour chaque caractère examiné, on met de côté tous les référents incompatibles. Quand tous les caractères ont ainsi été explorés, soit les référents restant en lice appartiennent à la même classe, et celle-ci devient la classe de détermination, soit

ils se répartissent dans plusieurs classes et la détermination est incomplète. S'il ne reste aucun référent, il y a une erreur quelque part, soit dans la description de l'indéterminé, soit dans celle des référents, soit dans l'affectation des référents aux différentes classes ; à moins qu'il ne s'agisse de quelque chose de nouveau, ne se rapportant à rien de connu.

1.7 Apprentissage et raisonnement

L'apprentissage est en lui-même une activité intelligente de l'être humain. Le but de l'apprentissage automatique effectué par une machine est de simuler l'apprentissage humain à l'aide de différents mécanismes de raisonnement.

Le raisonnement agit sur des connaissances dont on constate plusieurs niveaux de généralité : faits particuliers, définitions de concepts (règles), méthodes de résolution d'un problème, méta-connaissances, etc.. De plus, ces connaissances sont structurées dans notre cerveau selon un modèle. Pour être capable de simuler le raisonnement, il faut être en mesure de représenter ces différentes sortes de connaissances. On constate de même que ces connaissances évoluent avec le temps, dans le sens d'un enrichissement (espéré). Pour Michalski (1986), l'apprentissage est "lié à la construction ou modification des représentations de ce que l'on expérimente".

Si l'on veut doter les machines de capacités d'apprentissage, il faut absolument prendre en compte la définition d'une structure pour représenter l'espace des connaissances, ainsi que des moyens d'y accéder pour les modifier ou pour en générer de nouvelles. C'est la problématique de la représentation des connaissances.

Classiquement, les systèmes experts ont utilisé le formalisme des règles de production pour modéliser les connaissances d'un expert. L'acquisition des connaissances s'effectue par l'intermédiaire d'un cogniticien qui aide l'expert à expliciter ses règles de décision. Ensuite, l'apprentissage met en place un mode de raisonnement par **déduction** à partir de ces règles explicites et de faits nouveaux qui leur sont présentés. Le système expert infère des conclusions dont les résultats valides seront ajoutés dans la base de connaissances.

Nous considérons l'apprentissage comme le processus de classification (discrimination) qui permet de généraliser des cas spécifiques pour construire une définition abstraite (des règles de décision) en fonction d'un "bon" critère de classification. Il s'agit d'apprentissage où le raisonnement se fait d'abord par induction. Ensuite, comme pour les systèmes experts classiques, on déduit à partir de ces nouvelles connaissances qu'un nouveau cas est couvert par cette définition abstraite.

Les généralisations "de haut niveau" extraites à partir des cas sont utiles pour comparer des concepts différents, les valider les uns par rapport aux autres (notamment par rapport à ceux élaborés de manière classique), mais aussi pour identifier rapidement une nouvelle observation. Ce raisonnement nécessite donc une classification préalable.

Une autre forme de raisonnement logique, introduite par (Peirce 1965), est l'**abduction**. Elle est l'opération qui consiste à choisir une hypothèse explicative obtenue en faisant la trace arrière des règles du domaine, compte tenu des conclusions supposées vraies. Par exemple, soit la règle suivante (modus ponens) qui permet de déduire que si l'on observe du feu, alors on a de la fumée :

$$R : \forall x \in \{\text{lieux}\},\ \text{feu}(x) \Rightarrow \text{fumée}(x)$$

Dire qu'il n'y a pas de fumée sans feu, c'est faire de l'abduction : on fait l'hypothèse qu'il y a un feu du fait que l'on observe de la fumée et que l'on connait R. La déduction est le raisonnement inverse exprimé par la règle R. Pour l'induction, on doit observer qu'à chaque fois qu'il y a un feu quelque part, on observe aussi de la fumée à ces endroits, et on construit donc la règle générale R.

Une autre forme de raisonnement fait aujourd'hui l'objet de recherches actives : elle repose sur les exemples eux-mêmes sans chercher à les généraliser. L'idée consiste à interpréter une nouvelle observation à l'aide d'un cas similaire extrait du système et choisi comme guide (Baress & Porter 1990). C'est le principe du **raisonnement par cas** (RàPC ou *Case Base Reasoning*).

Raisonner consiste à comparer la proximité des cas avec la nouvelle observation par une mesure de distance. Il ne nécessite donc qu'un classement des individus au préalable (individus pré-classés par un nom de classe). Pour résumer, nous donnons la figure 1.6 suivante :

Raisonnement	Entrée	Sortie
déduction	prémisses + règles	concepts
induction	prémisses + classes	règles + concepts
abduction	règles + concepts	prémisses
"par cas"	prémisse + classes	classes

FIG. 1.6 – *Les modes principaux de raisonnement en apprentissage automatique.*

En définitive, l'aspect très important du raisonnement en apprentissage automatique doit être la mise en oeuvre concertée dans les algorithmes, de mécanismes symboliques logiques issus des recherches en intelligence artificielle (représentation des connaissances, règles de généralisation, stratégies de contrôle, etc.) et de méthodes numériques performantes (distances, mesures de proximité, entropie, etc.) propres à l'Analyse des Données et aux statistiques. Cette nécessité est à l'origine du développement des recherches sur le traitement des connaissances "symboliques - numériques" en apprentissage (Kodratoff 1991).

1.8 Individus, instances et objets

L'**individu** est considéré de manière extensive, synonyme d'un élément d'un groupe ou d'une classe. Dans l'idéal, un individu est un être réel, une entité tangible et distincte. Il s'agit d'un sujet unitaire correspondant à un spécimen en biologie. Seul un individu peut être décrit, et ce n'est que dans un sens généralisé que l'on peut parler de "description de classe". Dans ce contexte, l'individu est synthétique et correspond à un ensemble d'éléments distincts comme par exemple l'Espèce avec ses différents spécimens.

L'**instance** est l'individu passé, présent et à venir qui appartient à un concept (le petit chien à naître fait partie du concept de chien) alors que l'individu existe indépendamment de celui-ci. L'individu appartient à la classe, l'instance appartient au concept.

Du point de vue mathématique, l'individu fait partie d'une **population observable** notée Π que l'observateur cherche à décrire. Une fois observé, l'individu devient objet d'observation noté ω. Une fois décrit, l'objet possède une description notée $d(w)$. L'observateur ou le descripteur (celui qui décrit) s'est approprié l'individu (le sujet) qui est devenu un objet de description (observé ou décrit). La population **observée** est notée Ω.

L'**objet** prend différentes significations selon le point de vue et l'échelle d'observation auxquels l'observateur se place : du point de vue d'une "description de classe", l'objet est pris comme un élément de cette classe, c'est-à-dire un individu. Par contre, si l'on se place à l'échelle d'une description individuelle, l'objet correspond à un composant de l'individu (ou partie "individualisable"). Tout dépend donc du point de vue. Un objet est une donc entité descriptive d'un individu, un individu est une entité descriptive d'une classe.

Pour illustrer cette distinction, considérons l'ensemble (taxon) des Mammifères : en se plaçant du point de vue de la "description de cette classe", l'objet sera par exemple une baleine ou un éléphant particulier. Par contre, en considérant la description d'un individu de la classe des Mammifères, l'objet sera l'une des entités descriptives de cet individu, à savoir sa tête, son tronc, ses jambes, etc..

Dans cette thèse, nous nous plaçons dans le second cas de figure : nous souhaitons acquérir des descriptions d'individus dont les objets sont les différents composants de ces individus à analyser.

Entre individu et classe, la relation qui lie ces deux notions est celle d'appartenance de l'individu à la classe : l'individu ω est un élément de l'ensemble C. Par opposition, deux classes emboitées sont liées par la relation d'inclusion ensembliste.

1.8.1 Conclusion et poursuite de l'exposé

Nous avons présenté la terminologie et le sens parfois divergeant accordé aux différents concepts manipulés par les biologistes, les mathématiciens et les philosophes : classes, concept, individu, instance, objet, etc.. Nous avons également donné un aperçu des différentes formes de raisonnement : déductif, inductif, par comparaison, qui permettent de manipuler des connaissances.

Notre objectif est guidé par la conception d'un système informatique qui prenne en compte, dans la mesure du possible, les connaissances des experts en Systématique, en leur offrant la possibilité d'expliciter des connaissances descriptives et classificatoires, de les analyser et d'inférer de nouvelles connaissances.

Nous nous attachons, dans suite de notre exposé, à explorer les principaux modèles et formalismes de *représentation des connaissances*, ainsi que les raisonnements associés. En effet, la représentation des connaissances fait appel à divers formalismes pour organiser les connaissances, les stocker et offrir des opérations formelles qui facilitent leur manipulation et permettent aux machines de faire des conclusions sur les données et/ou objets étudiés.

Les formalismes que nous présentons sont issus principalement de deux grands courants de recherche en informatique, l'Analyse de Données et l'Intelligence Artificielle. Nos nous efforcerons de détacher de ces modèles, les formes de représentation qui sont les plus pertinentes pour notre problématique en indiquant leurs avantages et inconvénients.

1.9 l'Analyse de Données

L'analyse de données peut se définir comme un ensemble de méthodes dont le but essentiel est la mise en relief des relations existants entre les objets (appelés individus), entre les variables qui les caractérisent et entre les individus et les variables. Une analyse des données consistera surtout à situer globalement ces individus les uns par rapport aux autres, et à mesurer l'importance relative de chacune des variables dans cette répartition (Diday et al. 1982).

Parmi les méthodes de l'Analyse de Données, on distingue :

- Les méthodes de *l'analyse factorielle* qui reposent sur l'algèbre linéaire : les individus sont représentés dans un espace de dimension n, on cherche une représentation dans un espace réduit ($p < n$).
- Les méthodes de *l'analyse discriminante* : étant donné un ensemble d'individus Ω caractérisés par un ensemble de p variables descriptives et une partition *a priori* en k classes de Ω, le but est de séparer au mieux ces k classes à partir des p descripteurs.
- Les méthodes de *classification automatique* : il s'agit de regrouper les individus en plusieurs classes en fonction des variables qui les caractérisent. Ces classes peuvent être disjointes (partitions) ou liées entre elles pour former des structures telles que des partitions, des hiérarchies ou des pyramides.

1.9.1 Les données de l'Analyse de Données

Les données ou individus sont décrits par un ensemble de variables aussi appelées attributs ou descripteurs. Les variables correspondent à un ensemble de paramètres choisit par l'utilisateur pour décrire les individus. On distingue deux types de variables : les variables *quantitatives* (numériques) et les variables *qualitatives* (symboliques). Une variable quantitative peut être par exemple le poids d'une personne, son âge ou sa taille qui prend des valeurs pour lesquelles les opérations arithmétiques (ex. moyenne, différence, etc.) ont un sens. Une variable qualitative prend des valeurs symboliques (ex. nom, sexe, couleur des cheveux, etc.). Une forme particulière de variable symbolique appelée variable *booléenne* prend deux valeurs possibles 1 (présence de la caractéristique) ou 0 (absence de la caractéristique). Les valeurs que peut prendre une variable sont également appelées *modalités*.

Classiquement, l'ensemble des variables et individus est généralement organisé sous la forme d'un tableau de données représenté par une matrice de n lignes et p colonnes, les lignes du tableau correspondent aux individus et les colonnes du tableau aux variables (fig. 1.7 page ci-contre).

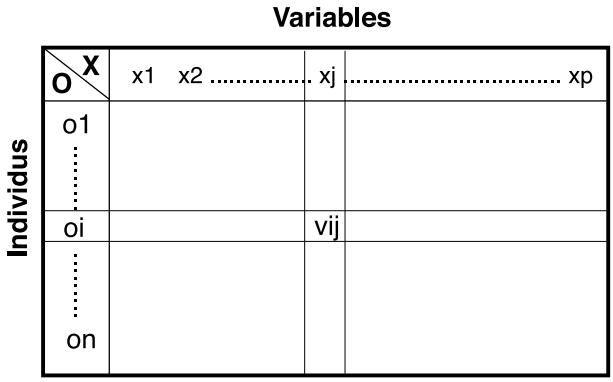


FIG. 1.7 – *Exemple de tableau de l'analyse de données. O désigne l'ensemble des individus, X l'ensemble des variables et v_{ij} désigne la valeur de l'individu o_i pour la variable x_j.*

1.9.2 Types de tableaux traités par l'Analyse de Données

Différents types de tableaux sont considérés en Analyse de Données :

- les tableaux de données quantitatives : toutes les variables sont numériques, chaque individu est représenté par un vecteur de $\mathbb{R}^p$.
- les tableaux binaires : les individus sont décrits uniquement à l'aide de variables qualitatives binaires.
- les tableaux de modalités : chaque variable peut prendre un ensemble de modalités ordonnées ou non ; chaque cellule $x_i^j = y_j(o_i)$ contient un nombre entier correspondant à l'une des modalités de la variable y_j.
- les tableaux de contingence ou de fréquence qui enregistrent le nombre d'occurrences simultanées d'un individu avec chacune des propriétés qui le décrit. La valeur de chaque variable associée à un paramètre est donc le nombre d'occurrences de ce paramètre.
- les tableaux hétérogènes : les variables sont de différents types (quantitatives, qualitatives, etc.).

Dans plusieurs cas, lorsque le tableau de données est hétérogène ou plus généralement lorsque le type du tableau est incompatible avec la méthode d'analyse que l'on cherche à appliquer, on applique un pré-traitement qui consiste à effectuer un changement de variable par codage (Jambu 1999). Les principaux types de tableaux recodés sont : les tableaux de données quantitatives *centrés* et *centré-réduits* ; les tableaux de modalités mis sous forme de tableaux binaires. Dans ce dernier cas, chaque modalité d'une variable est codée par une nouvelle variable booléenne.

1.9.3 Méthodes de classification

Parmi les méthodes d'analyse de données, les méthodes de classification tiennent une place particulièrement importante.

Nuées dynamiques

La méthode des **nuées dynamiques** (Diday et al. 1982) par exemple consiste à rechercher une partition en k classes bien agrégées et bien séparées entre elles d'une population d'individus représentés par des vecteurs de $\mathbb{R}^p$. Cette méthode nécessite au préalable de définir un mode de représentation de tout groupe d'individus : le *noyau* (ou prototype) du groupe.

La méthode consiste à estimer ou à tirer au hasard k noyaux, puis à affecter chaque individus de la population à un noyau au moyen d'une fonction d'affectation. On obtient ainsi k classes d'individus dont on recalcule les noyaux. On réitère ensuite l'opération avec les nouveaux noyaux jusqu'à obtenir une partition stable. La convergence vers une partition stable est assurée sous certaines conditions portant sur les fonctions d'affectation et la représentation.

La méthode des nuées dynamiques produit une partition de l'ensemble des objets en un ensemble de classes disjointes, mais elle nécessite de fixer *a priori* le nombre de classes.

Classification ascendante hiérarchique

A partir d'un ensemble d'individus, la classification hiérarchique ascendante construit un ensemble de classes qui forment des partitions hiérarchiquement emboîtées. La méthode repose sur la définition d'un indice d'aggrégation δ, généralement construit à partir d'un indice de dissimilarité ou de distance entre les individus.

A partir d'une partition initiale dont les classes sont réduites à un élément (un individu), on construit une nouvelle partition en réunissant les deux classes qui minimisent l'indice d'aggrégation δ. On réitère le procédé sur la nouvelle partition ainsi obtenue jusqu'à obtenir une partition ne contenant qu'une seule classe. Une hiérarchie est une structure plus riche qu'une partition puisqu'elle définit une suite de partitions. Le nombre de classe n'est de plus pas connu à priori.

Classification

La classification pyramidale constitue une extension du modèle hiérarchique et permet d'obtenir des paliers dont l'intersection n'est pas nécessairement vide. Le principe de construction s'inspire de celui des hiérarchies, la différence se situe à l'étape d'agrégation. La partition initiale est constituée des classes ne contenant qu'un seul élément. On calcule alors une nouvelle partition en agrégeant les deux classes les plus proches qui n'ont pas encore été agrégées deux fois. Puis on procède de manière itérative jusqu'à l'obtention d'une seule classe contenant tous les objets. Une classe peut donc appartenir à deux agrégations, ce qui autorise la possibilité pour un palier d'appartenir à deux classes.

1.9.4 Limites des représentations de l'analyse de données

Le modèle des tableaux numériques de l'analyse de données « classique » présente un certain nombre de limitations pour le traitement de données complexes. Le modèle tabulaire traite des observations individuelles et ne prend pas en compte la disjonction ou la conjonction de valeurs pour exprimer respectivement la variation d'imprécision ou la présence simultanée de plusieurs valeurs pour un descripteur. Il ne permet pas de représenter des objets complexes dont

la description contient des relations avec d'autres objets. Le traitement de telles données nécessite alors de réduire l'information au cadre tabulaire numérique par des codages de l'information, ce qui nécessite un compromis entre la perte d'information acceptée et l'alourdissement de l'analyse dû à la multiplicité des variables introduites.

1.9.5 L'Analyse de Données Symbolique

Le besoin de traiter des données plus complexes et donc plus générales a conduit au développement d'une approche symbolique de l'analyse de données (Diday 1989). Ce formalisme a été repris depuis par de nombreux chercheurs en analyse des données numériques-symboliques. En particulier, l'extension des objets symboliques au traitement des logiques modales (Diday 1991) a abouti à une modification de leur définition.

Représentation des objets symboliques

Un objet symbolique est un modèle mathématique d'un « concept » (Diday & Emilion 1996). Il est défini par une **intension**, c'est-à-dire l'ensemble des propriétés qui doivent être satisfaites par un individu pour appartenir au concept et un moyen de calculer son **extension**, c'est-à-dire l'ensemble des individus qui satisfont l'intension. Un objet symbolique a est dit *plus général* (resp. plus spécifique) qu'un objet symbolique b, si l'extension de a contient (resp. est contenue par) l'extension de b.

Soit $D = \{\omega_1,...,\omega_n\} \subseteq \Omega$ un ensemble d'individus (les cas) contenus dans un tableau de données symboliques ou une base de données. Ω désigne l'ensemble des cas observables (non nécessairement connus ou accessibles). Chaque individu ω est caractérisé par un ensemble de variables (appelés également attributs ou descripteurs) $Y = \{y_1, \ldots, y_p\}$. Dans un tableau de données symboliques, chaque ligne du tableau correspond à une description symbolique (un objet symbolique) ω, et chaque colonne à une variable y.

Chaque cellule du tableau peut contenir des données de différents types :

- valeur *quantitative* simple : par exemple le poids ou la taille d'un individu ω, $Poids(\omega) = 67$,
- valeur *qualitative* simple : par exemple $Ville(\omega) = Paris$,
- valeur *multiple* (ou ensemble de valeur) qui peut être qualitative ou quantitative. Par exemple $Poids(\omega) = \{60,67,75\}$ signifie que le poids de l'individu est 60, 67 ou 75 kg, ou $Ville(\omega) = \{Paris,\ Londres,\ Amsterdam\}$ désigne que la variable $Ville$ pour ω est $Paris$, $Londres$ ou $Amsterdam$. Notons que les cas 1 et 2 sont des cas particulier de valeur multiple.
- valeur *quantitative intervalle* : par exemple $Poids(\omega) = [60,75]$ pour signifier que le poids de ω se situe entre 60 et 75 kg.
- *histogramme* ou fonction d'appartenance.

Apport de l'Analyse de Données Symbolique

Ce formalisme contient et généralise le modèle classique tabulaire. L'apport des objets symboliques peut se résumer par les aspects suivants :

- Un objet symbolique est une description en compréhension d'une classe d'objets élémentaires qui en constituent l'extension, on atteint ainsi un niveau conceptuel de description de données.
- Les objets symboliques permettent de représenter des données pour lesquelles chaque variable peut prendre des valeurs multiples pour un même objet et tient compte des liens qui peuvent éventuellement exister entre les valeurs des variables.
- Les objets symboliques permettent de représenter des données où certaines variables peuvent être *inapplicables* et les variables prendre la valeur *inconnue*.
- Un objet symbolique peut être généralisé ou spécialisé de façon à étendre ou restreindre son extension.
- Les opérations de généralisation et de spécialisation peuvent tenir compte d'une taxonomie, c'est-à-dire que les valeurs peuvent être organisées suivant une relation d'ordre.
- L'aspect déterministe peut-être atténué par l'utilisation d'objets modaux qui permettent de représenter l'incertitude quant à la valeur d'une variable utilisée pour décrire un objet symbolique (Lebbe 1991).
- L'Analyse de Données Symbolique, en généralisant les représentations tout en utilisant les méthodes de l'Analyse de Données « classique » permet donc de manipuler des objets plus complexes et d'obtenir des classes définies à la fois en extension et en compréhension et donc plus facilement interprétables.

1.9.6 Discussion

Le formalisme des objets symboliques a été adopté pour la formalisation des connaissances en biologie par Conruyt (1994), pour la représentation des *descriptions* d'objets biologiques (spécimens) sous la forme de *Hordes Symboliques*. Cette formalisation a abouti au développement du système Hyperquest et une application sur la représentation de spécimens d'*Eponges marines* a été validé par les experts.

Parallèlement, une formalisation des concepts de la Systématique a été effectuée par (Lebbe 1991, Vignes 1991, Vignes 2000) dans le formalisme des objets symboliques pour la représentation des *taxons* d'un domaine considéré. Les objets biologiques (spécimen) sont alors représentés en *intension* par des objets symboliques, qui donnent ainsi le moyen de calculer leur *extension*.

Dans notre recherche d'un modèle général, unificateur, de représentation des connaissances en Sciences de la vie[29], nous souhaitons intégrer les deux approches, c'est-à-dire permettre simultanément de représenter des descriptions de spécimens en tant qu'instances de concepts (les taxons) modélisés et organisés sous forme de hiérarchies. L'Intelligence Artificielle apporte également des réponses pertinentes pour la structuration hiérarchiques des connaissances, et les formalismes les plus achevés, les *logiques terminologiques* (LT) et les *Systèmes de Représentation*

29. L'exposé du modèle CoDesc fera l'objet du chapitre suivant.

des Connaissances à Objets (SRCO) nous ont beaucoup inspiré pour l'élaboration de notre modèle. Nous en présentons ici les principaux développements, depuis les *réseaux sémantiques* sur lesquels se fondent les *schémas*, jusqu'aux modèlex à objets, qui manipulent des instances et des classes selon différents *points de vue* et offrent un ensemble de mécanismes d'inférence (filtrage, classification, etc.) pour les manipuler.

1.10 Les modèles hiérarchiques de l'Intelligence Artificielle

1.10.1 Les réseaux sémantiques

Les réseaux sémantiques mis en œuvre par Quillian (1968) ont été conçus à l'origine comme un modèle psychologique explicite de la mémoire associative humaine. La mémoire est vue comme un réseau d'unités d'information, activées par un mécanisme qui propage des signaux à travers le réseau, appelé "procédure d'activation". En Intelligence Artificielle, le premier système à utiliser les techniques des réseaux sémantiques, SIR (Raphael 1968), répond à des questions qui demandent un raisonnement simple, avec des prédicats binaires. Plusieurs variétés de réseaux sémantiques émergent par la suite, à la fin des années 70 : les réseaux partitionnés de Hendrix (Hendrix 1979) qui permettent de manipuler des assertions comme la logique, les réseaux de propagation de marqueurs (Fahlman 1979) et les réseaux procéduraux (Levesque & Mylopoulos 1979) issues des idées des représentations procédurales de Winograd (1972).

Représentation des connaissances

Les réseaux sémantiques sont fondés sur un modèle de graphe permettant de combiner la représentation des concepts par l'intermédiaire des nœuds et des relations entre concepts, par des arcs orientés. Les étiquettes sur les arcs spécifient le type de la relation entre deux concepts. Ces relations correspondent par exemple à des liens de causalité, des relations spatiales ou temporelles ou encore des relations de spécialisation ou de compositions entre concepts. Deux relations particulières offrent la possibilité de structurer la connaissance en une hiérarchie de concepts, la relation sorte-de entre deux concepts génériques et la relation est-un entre un concept générique et un concept individuel. Un concept générique correspond à une définition en intention, un concept individuel à une description en extension. Les arcs d'héritage introduisent une organisation hiérarchique des concepts qui favorise une factorisation des informations au sein d'un réseau et facilite le raisonnement. En effet, un concept qui spécialise un autre possède toutes les propriétés et relations de celui-ci. Le principe de structurer les connaissances en hiérarchies est à l'origine de toute une famille de systèmes, les systèmes hiérarchiques et en particulier les logiques terminologiques.

Un ensemble d'opérations associé au réseau permet de l'exploiter : création et suppression de nœuds et d'arcs, parcours du graphe des relations et filtrages. D'autres types de raisonnement se sont constitués, exploitant la structure graphique des réseaux et sans égard à la nature des arcs parcourus. Le manque de sémantique claire des éléments d'un réseau (Woods 1975) (les arcs, les liens d'héritage) a été à l'origine des études poussées destinées à définir un cadre formel pour le modèle (Brachman 1983). Celles-ci ont profité essentiellement au développement d'autres formalismes de représentation, tels ques les graphes conceptuels, les logiques de description et la

représentation par objets.

Raisonnement

Deux méthodes sont couramment utilisées dans les réseaux sémantiques, l'inférence par héritage et le filtrage. Le filtrage consiste à parcourir le graphe et à chercher tous les sous-graphes ayant des propriétés ou une structure commune avec un graphe cible. Cette recherche correspond à un appariement de graphes. Ainsi, par exemple, une interrogation est modélisée par un sous-réseau où les nœuds représentant l'information recherchée sont étiquetés par des variables qu'une procédure de filtrage tente de lier avec des nœuds du réseau traité (Masini et al. 1989).

L'héritage peut intervenir dans le filtrage. Les concepts spécialisés (concepts individuels) récupèrent les informations contenues dans les concepts plus généraux (concepts génériques) par un parcours des liens de spécialisation est-un (ou lien isa). Dans les réseaux à propagation de marqueurs (Fahlman 1979) du système NETL, un mécanisme d'inférence basé sur la propagation de marqueurs associés aux nœuds du réseau, permet de parcourir le graphe d'héritage et de récupérer les propriétés les plus spécifiques d'un nœud donné. NETL fut un des premiers systèmes de représentation permettant de traiter les exceptions, par l'intermédiaire de liens d'exception.

Intérêt et limites des réseaux sémantiques

L'apport des réseaux sémantique est surtout d'avoir introduit la structure dans les représentations, tout en restant propositionnels comme la logique et les systèmes à base de règles. Le type de représentation sous forme de graphe rend visible les relations entre les objets et s'adapte bien aux domaines où les concepts sont simples et fortement liés, comme le langage naturel. De plus, la notion de distance entre concepts peut être calculée à partir du nombre de liens composant le chemin liant deux concepts. Les diverses opérations possibles sur ces réseaux, telles que la recherche d'information par filtrage et les mécanismes d'inférence par héritage, permettent de manipuler les connaissances et de répondre à des questions simples sur le réseau.

Les limites du modèle proviennent essentiellement du manque de sémantique formelle associée aux nœuds et aux relations. Même si les relations de subsomption (relation sorte-de) et les liens d'instanciation (lien est-un) sont traitées de manière adaptée, tout autre relation est traitée indépendamment de la sémantique donnée par le simple label. Ce problème est développé par Woods (1975) dans l'article « What's in a link? » et corrigé par Brachman & Schmolze (1985*a*) dans le système *KL-ONE* qui développe une sémantique formelle basée sur la logique et la relation de subsomption. Les réseaux sémantiques offrent une représentation statique du monde, ce qui rend difficile la modélisation d'information évolutive.

1.10.2 Les schémas

Les *schémas* (ou *frames*) sont issus des travaux en représentation des connaissances de Minsky (1975) qui s'est inspiré de théories de psychologie, comme celle des schémas (Bartlett 1964) et celle des paradigmes (Kuhn 1983) ainsi que de la théorie des prototypes (Rosch & Lloyd 1978). Il souhaitait pouvoir rendre compte de certains aspects du raisonnement humain tels que la réutilisation des expériences passées ou l'inférence de nouvelles caractéristiques des entités

représentées, ce qui est difficile à réaliser par un formalisme propositionnel comme la logique ou les réseaux sémantiques. L'unité de représentation des connaissances est le schéma, structure dynamique représentant des situations prototypages comme par exemple conduire une voiture, aller au restaurant, etc. En tant que prototype d'un ensemble de situations, le schéma a des informations générales valides pour toutes les situations et des informations spécifiques à chaque situation pour lesquelles il a une valeur par défaut, la plus probable.

Dans le cadre de cette théorie, la rapidité des activités mentales humaines s'explique par le fait qu'une situation nouvelle est identifiée dans la mémoire à la situation prototypique la plus proche, les informations générales de ce prototype sont ensuite récupérées et éventuellement modifiées pour créer la représentation précise de la nouvelle situation.

Les schémas sont regroupés en systèmes de schémas. Un schéma peut être transformé en un autre par une opération de transformation qui copie la relation représentée par le système.

Représentation des connaissances

Un schéma est une structure de données à trois niveaux *schéma-attribut-valeur* qui représente un objet ou une situation typique comme conduire une voiture ou chanter un air de musique. Cette structure représente une unité d'information descriptive (un descripteur) avec une sémantique donnée (Winston 1977, Bobrow & Winograd 1977, Masini et al. 1989). Si le descripteur est un objet, il peut aussi bien représenter une famille d'objets (une classe) qu'un objet particulier (une instance de la classe). La distinction entre ces deux types d'objets est importante du point de vue de l'héritage (qui permet le partage et la réutilisation des propriétés entre objets) car la nature des relations qu'ils entretiennent n'est pas la même :

- Deux objets de type classes sont reliés par la relation d'inclusion entre ensembles avec un lien de type sorte-de,
- Un objet de type instance est un élément appartenant à un objet de type classe et le lien est de type est-un.

Raisonnement

Les représentations par Schémas offrent deux types de mécanismes de raisonnement : les mécanismes globaux qui travaillent sur toute la base et les mécanismes locaux qui font des inférences au niveau d'un schéma ou d'un attribut d'un schéma. Les mécanismes globaux sont *l'héritage*, *le filtrage* et *la classification*. Les mécanismes locaux sont les *réflexes* de contrôle et de calcul. La relation de spécialisation représente l'inclusion ensembliste : toute les instances d'une classe sont aussi instances des super-classes. Elles héritent donc des propriétés décrites dans les super-classes. Le mécanisme d'héritage dynamique des propriétés permet la récupération de ces informations à travers les liens de spécialisation et évite ainsi la recopie explicite des attributs. L'héritage dynamique permet de garantir que toute modification faite à une classe est prise en compte automatiquement par ses sous-classes.

Le mécanisme de filtrage recherche les schémas satisfaisant certaines caractéristiques données par un schéma particulier appelé *filtre*, par un mécanisme d'appariement avec les schémas présents dans la base. Le filtrage se distingue d'une requête classique de bases de données par

plusieurs aspects. D'une part, les instances d'une base de connaissances pouvant être incomplètes, le résultat du filtrage n'est pas une liste fixe d'instances, mais une liste de schémas d'instances où chaque schéma peut-être complété de différentes manières et produire diverses instances. D'autre part, grâce à l'organisation hiérarchique des schémas de classes, l'ensemble des schémas d'instances pouvant répondre au filtre est vite réduit à ceux dont les classes d'appartenance ne sont pas en contradiction avec les contraintes du filtre. Enfin, l'étape d'appariement peut faire appel aux mécanismes de raisonnement de bas niveau comme les réflexes et l'attachement procédural pour un calcul. Dans certains systèmes comme SHIRKA (Rechenmann 1985, Rechenmann 1988), le filtrage peut être un mécanisme local, une facette d'un attribut d'une classe.

La classification consiste à positionner un nouveau schéma dans une hiérarchie de schémas existante. En général, les systèmes de schémas distinguent la classification de classes de la classification d'instances. La classification de classes modifie les liens taxinomiques des classes et constitue un mécanisme de maintient et de gestion de la base. La classification d'instances est un mécanisme qui permet de compléter la connaissance d'une nouvelle instance en la plaçant correctement dans la base et en récupérant l'information déduite de ce classement. Le mécanisme de classification d'instances correspond à une stratégie de résolution de problèmes.

Discussion

Les schémas sont très utiles pour modéliser les domaines où les objets sont complexes et riches. Le regroupement de toutes les informations d'un objet dans une unité fournit une représentation structurelle concise et facilement exploitable. La structuration de la connaissance en de telles unités et les liens entre ces unités facilitent la manipulation des connaissances.

Cependant, les systèmes qui manipulent les Schémas manquent généralement d'un formalisme de base, ce qui entraîne un manque de rigueur. Par ailleurs, les systèmes de schémas ne font pas la distinction entre les caractéristiques définitionnelles d'une classe : celles qui doivent être satisfaites par les instances pour appartenir à la classe, et les caractéristiques déduites : celles qui sont vraies pour les instances de la classe (Brachman 1983). Cette distinction est un des grands apports des langages terminologiques.

1.10.3 Les logiques terminologiques

Les idées développées dans le système KL-ONE par Brachman dès 1978 (Brachman & Schmolze 1985*a*) ont donné naissance à un courant de recherche très actif qui est à l'origine d'une famille de langages de représentation appelés *logiques de descriptions* ou *logiques terminologiques* (Nebel 1990, Napoli 1997). Les logiques terminologiques ont pour base les langages de frames et les réseaux sémantiques. Ils sont aussi appelés systèmes hybrides, systèmes de représentation de connaissances basés sur la classification (CBS) ou encore langages de subsomption de termes (TSL).

Les logiques terminologiques mettent l'accent sur la structure dans la description des connaissances génériques. Elles offrent des langages permettant de décrire les concepts d'un domaine (la terminologie) à l'aide de termes structurés. Ceux-ci peuvent être construits à partir d'autres termes à l'aide de connecteurs. Ils sont organisés dans une hiérarchie selon la relation de subsumption.

Représentation des connaissances

Dans les logiques de descriptions, un concept permet de représenter une classe d'individus, tandis qu'un rôle représente une propriété associée à un concept. Un concept correspond à une entité générique d'un domaine d'application, un rôle à une relation binaire définie entre concepts, et un individu à une entité particulière, instances du concept.

- Un concept et un rôle possèdent une description structurée, obtenue à partir d'un certain nombre de constructeurs. Une sémantique est associée à chaque description de concept et de rôle par l'intermédiaire d'une interprétation . Les opérations sur les concepts et les rôles sont réalisées en accord avec cette sémantique.
- Les connaissances sont prises en compte selon leur niveau de généralité : la représentation et la manipulation des concepts et des rôles relèvent du niveau terminologique ; la description et la manipulation des individus relèvent du niveau factuel ou niveau des assertions.
- La relation de subsomption permet de comparer deux concepts (ou deux rôles) entre eux, afin de déterminer lequel est plus général que l'autre. Si un concept C subsume un concept D, alors le subsumant C est plus général que le subsumé D et D partage des propriétés avec C. Une base de connaissance se compose alors d'une hiérarchie de concepts et d'une hiérarchie de rôle. L'accès aux éléments de la base se fait par l'intermédiaire de primitives qui permettent d'associer une valeur à un rôle (écriture), de modifier une valeur, et de retrouver la valeur d'un rôle (lecture).
- Les opérations qui sont à la base du raisonnement sont la classification et l'instanciation. La première s'applique aux concepts et aux rôles, et permet de déterminer l'ensemble des ascendants directs d'un concept dans la hiérarchie des concepts (subsumants les plus spécifiques ou SPS). L'instanciation peut-être vue comme une identification et permet de retrouver les concepts dont un individu est susceptible d'appartenir.

Les connaissances sont prises en compte selon deux niveaux. La représentation et la manipulation des concepts et des rôles relèvent du niveau terminologique appelé *Tbox*. La description et la manipulation des individus relèvent du niveau factuel aussi qualifié de *Abox*. Les concepts peuvent être primitifs lorsqu'ils dénotent des catégories naturelles comme les personnes, les animaux, etc. Leur description est alors incomplètement spécifiée et leurs rôles sont nécessaires : il n'est pas possible d'affirmer qu'un individu est un représentant d'un concept primitif en examinant ses seuls rôles. Les concepts primitifs servent de base à la construction des concept définis dont la description est complètement spécifiée ; les rôles dénotent cette fois une condition nécessaire et suffisante.

Discussion

Le langage de définition des logiques terminologiques introduit des notions qui sont souvent difficile à comprendre, ce qui rend le développement de la base plus complexe. C'est le cas des distinctions entre propriétés définitionnelles et propriétés contingentes, entre propriétés nécessaires et propriétés suffisantes et entre subsomption extensionnelle et subsomption intensionnelle, comme le souligne (Drews 1993). Un aspect particulièrement difficile à comprendre est la distinction entre le raisonnement terminologique et le raisonnement assertionnel. En effet, une affirmation du style "ce téléphone est rouge" peut être interprétée comme une assertion

sur un individu particulier du concept "téléphone", indiquant qu'il est rouge, ou bien comme une assertion affirmant l'existence d'un individu du concept "téléphone rouge". Dans le premier cas, la définition du concept n'inclut pas la couleur : la propriété couleur ne fait pas partie de la définition, c'est une propriété contingente. Dans le deuxième cas, seuls les téléphones rouges satisfont le concept, la couleur fait partie de la condition d'appartenance au concept.

1.10.4 Les graphes conceptuels

L'objectif initial de la création de graphes conceptuels est la représentation intelligible des assertions en langage naturel (Sowa 1984, Sowa 1991). Les graphes conceptuels étaient destinés initialement à pallier les inconvénients de la logique classique. A savoir, le manque de structure dans les descriptions et dans l'organisation d'une base de connaissances. En même temps, les graphes conceptuels s'inspirent fortement des réseaux sémantiques : on peut les voir comme une extension de ceux-ci mais avec une syntaxe et une sémantique plus claire.

Représentation des connaissances à l'aide des graphes conceptuels

Un graphe conceptuel (Nogier 1991) est un graphe fini, biparti et connexe. Les deux types de nœuds qui constituent le graphe sont un ensemble de *concepts* et un ensemble de *relations conceptuelles* :

- Les concepts correspondent à des contenus de la pensée et obéissent à la notation suivante : [<type> : <référent>].
  Le type du concept détermine une structure pour le concept : si le concept est bien formé, il est "conforme" au type. Le référent du concept précise son sens, il donne le degré de quantification du concept. Les concepts d'un graphe peuvent être génériques ou individuels : les concepts génériques correspondent aux variables en logique et représentent des individus non spécifiés et les concepts individuels correspondent aux constantes de la logique et décrivent un individu particulier. Notons qu'un concept peut former à lui seul un graphe conceptuel.
- Les relations conceptuelles symbolisent les liens qui existent entre les concepts et sont représentées par la notation suivante : -> (. . .) ->
  De toute relation part ou arrive au moins un arc, chacun de ces arcs devant être lié à un concept. Si une relation a n arcs, ses arcs reçoivent les étiquettes 1,2,. . . ,n.

Un graphe conceptuel représente une seule formule logique (Sowa 1984) alors qu'un réseau sémantique représente une collection (ou produit) de formules et décrit leurs connections mutuelles en les immergeant dans leur contexte, le domaine du discours. Voir pour exemple la fig. 1.8 et son équivalent en notation graphique, fig. 1.9.

[Décoller]->
(Agent)->[Personne : Philippe]->(Caractéristique)->[Expérimenté]
(Objet)->[Parapente]

FIG. 1.8 – *Exemple de graphe conceptuel sous une représentation linéaire correspondant à la phrase « Philippe est expérimenté et décolle en parapente ».*

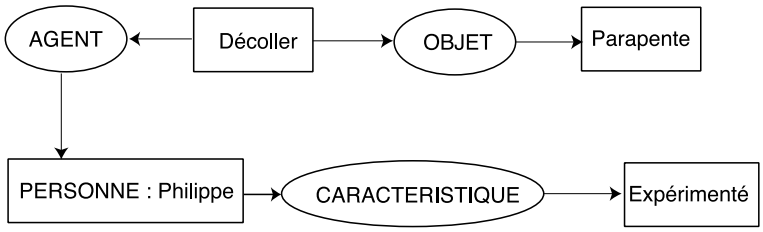


FIG. 1.9 – *Exemple de graphe conceptuel en notation graphique.*

Les concepts sont organisés en une *hiérarchie de concepts* (fig. 1.10) qui permet au modèle sémantique de généraliser et de spécialiser les concepts. Les concepts peuvent donc être représentés à différents niveaux d'abstraction. La hiérarchie de concepts est composée de diverses familles de concepts de même type, c'est à dire dépendant du même *hyperonyme* (Aimeur 1994). Par exemple le concept Aéronef est un hyperonyme de Avion, Planneur, ULM, etc. Ces différents concepts sont de type d'Aéronef.

La hiérarchie des concepts comporte deux autres types de concepts qui assurent la complétude du système et confère une structure de treillis à la hiérarchie des types. L'élément maximal de la relation d'ordre partiel de typage est le type universel $\top$ et comme élément minimal, le type absurde $\bot$. Le treillis de types permet d'inférer des relations de conformité entre un concept et un type. Ainsi, tout individu conforme à un type t l'est aussi à tous les sur-types de t dans le treillis, y compris le type universel. Si un concept est conforme à deux types s et t, alors il l'est aussi au type intersection $s \cap t$. Aucun concept n'est conforme au type absurde.

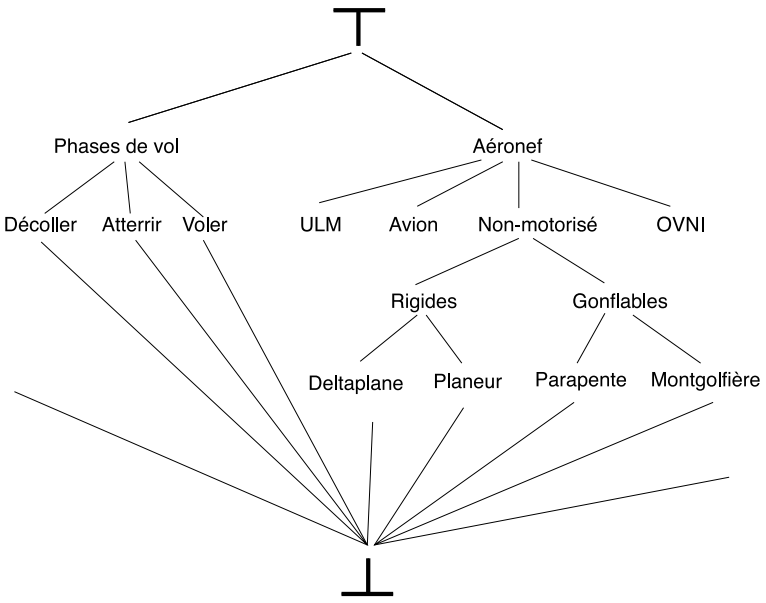


FIG. 1.10 – *Exemple de hiérarchie conceptuelle.*

Tout graphe conceptuel est créé à partir de *graphes canoniques*, les graphes primitifs de représentation. Ces graphes ont une sémantique associée et sont dits corrects pour la représentation. Pour éviter la génération des graphes absurdes, tout système de graphes conceptuels doit partir d'une base initiale de graphes canoniques et en générer d'autres à partir de quatre règles de

formation :

- La **copie** consiste à faire une copie stricte du graphe.
- La **restriction** est le remplacement du type d'un concept par un de ses sous-types. Par exemple, la fig. 1.11 illustre le remplacement du type Personne par le type Pilote, plus spécialisé.
- La **jointure** (ou le joint) permet de fusionner deux graphes ayant une partie commune en enlevant la partie commune d'un des deux graphes.
- La **simplification** enlève les relations dupliquées du graphe, ainsi que les arcs correspondants.

[Décoller]->
(Agent)->[Pilote : Philippe]->(Caractéristique)->[Expérimenté]
(Objet)->[Parapente]

FIG. 1.11 – *Exemple de restriction de type. Le type Personne de l'exemple 1.8 est remplacé par le type Pilote.*

Les règles de formation canonique (Tayse et al. 1990) sont des règles de spécialisation. La relation inverse, la généralisation est réflexive, antisymétrique et transitive et définit un ordre partiel sur le graphe, la *hiérarchie de généralisation*. La relation de généralisation est conservée par la relation de sous-typage de concepts, la relation d'individualisation d'un concept générique et la relation de sur-graphe :

- **Sous-type** : si le graphe u est identique au graphe v sauf pour quelques types de v qui ont été changés par des sous-types dans u, alors $u \leq v$.
- **Graphe universel** : Le graphe [T] est une généralisation de tous les graphes conceptuels.
- **Individus** : Si u est identique à v excepté que quelques concepts génériques de v ont été remplacés par des concepts individuels (conformes aux types) dans u, alors $u \leq v$.
- **Sous-graphe** : si v est un sous-graphe de u alors $u \leq v$.

La correspondance entre la généralisation de graphes et l'implication en logique des prédicats garantit que toute affirmation valide pour un graphe de la hiérarchie de généralisation, est aussi valide pour ses généralisations.

Outre les types simples, il est possible de définir des types complexes correspondant à tout un graphe conceptuel. Une telle définition consiste à donner une étiquette à un graphe en faisant ainsi une abstraction du graphe. A tout moment, des mécanismes de contraction et d'expansion permettent le passage du graphe à l'étiquette et de l'étiquette au graphe.

Raisonnement

Les graphes conceptuels offrent deux types de raisonnement qui correspondent à peu près aux approches définitionnelles et prototypiques des schémas, le raisonnement exact et le raisonnement plausible.

Le raisonnement exact utilise l'équivalence entre la logique du premier ordre et les graphes canoniques, ainsi que le hiérarchie de généralisation, pour inférer des connaissances logiquement correctes et sémantiquement convenables.

Le raisonnement plausible introduit les notions de schémas et de prototype pour inclure des connaissances spécifiques du domaine dans le raisonnement. Un schéma présente les concepts et relations couramment associés à un type de concept ; les différents schémas associés à un type de concept correspondent aux différentes façons de voir l'utilisation de ce type de concept. A la différence des définitions de types, les schémas n'établissent pas de conditions nécessaires et suffisantes pour leurs types. Si les schémas prennent en compte la connaissance du domaine au niveau des types de concepts, les prototypes le font au niveau des individus. Un prototype est un individu typique d'un concept, ayant des valeurs par défaut pour les différentes relations intervenant dans le concept. Les schémas et les prototypes permettent de raisonner sur des graphes incomplets mais le raisonnement, à la différence de la logique du premier ordre, n'est pas monotone, et les conclusions obtenues à un moment donné peuvent être remises en question plus tard.

Discussion

A la différence des réseaux sémantiques qui représentent dans un même réseau plusieurs propositions, ainsi que des définitions de concepts, des relations de sous-typages et des liens vers des perceptions du monde, les graphes conceptuels organisent les différents types d'éléments dans des structures différentes : la hiérarchie de spécialisation, le treillis de type. Cette distinction permet la manipulation et la vérification de chaque structure à partir d'un ensemble réduit de règle de cohérence.

D'autre part, l'équivalence entre les graphes conceptuels et la logique du premier ordre garantit que les inférences du modèle sont correctes. L'abstraction des concepts offre un mécanisme utile de définition de concepts complexes.

Cependant, le choix d'un graphe conceptuel comme unité de représentation pose les problèmes déjà soulignés pour la logique et les réseaux sémantiques. La connaissance concernant un objet est répartie dans différents graphes, ce qui pose de sérieux problèmes dès lors que le domaine possède des objets complexes et que la base est importante.

1.11 L'approche objets

1.11.1 Les objets

La notion d'objet est présente dans de nombreux domaines de l'informatique qui construisent des modèles conceptuels du monde réel, tels que les bases de données, le génie logiciel et l'intelligence artificielle. Cette notion est née d'une volonté d'avoir une représentation informatique de haut niveau qui reflète directement des entités du monde réel et sur laquelle raisonnent les utilisateurs humains. En programmation, les mécanismes de base sont maintenant bien connus et maîtrisés (cf. par exemple (Perrot & Napoli 1996). En Intelligence Artificielle, les langages à objets sont devenus omniprésents, bien que le terme objet revête des significations bien différentes suivant les communautés (Pachet 1997). On distingue traditionnellement deux familles de langages à objets : les langages de programmation par objets et les langages centrés-objets, appelés aussi représentaton par objets.

Les langages orientés objet

Les *langages orientés objets (LPOO)* sont dédiés à la programmation. Ils permettent de modéliser l'univers d'une application à l'aide d'une hiérarchie de classes. Le langage Simula (O.J.Dahl & Nygaard 1966) basé sur le langage Algol-60 a développé les idées qui ont conduit au premier langage objet : Smalltalk (Goldberg & Robson 1983). De nombreux langages se sont développés par la suite, en introduisant la notion de classes dans les langages existants tel que C++ (Stroustrup 1989), la version orientée-objet du langage C ou les versions objets du langage Lisp tel que CLOS (Bobrow & Stefik 1988*a*, Keene 1989). On peut également citer Eiffel (Meyer 1990) ou encore Java, qui fait référence à l'heure actuelle comme langage portable de conception orienté objets, en particulier pour le développement d'applications *on-line*.

Ces langages sont basés sur le paradigme **classe/instance** qui reposent sur la notion d'objet comme entité regroupant une *structure* et un textitcomportement, définis dans la classe de l'objet et donc communs à toutes les instances. La structure représente *l'essence* de l'objet, définisse de quoi est fait l'objet. Cette structure permet d'organiser les objets en un réseau par l'intermédiaire de relations. Le comportement d'un objet définit *ce que sait faire* l'objet. Il est représenté par un ensemble de méthodes (fonctions ou procédures), déclenchées par des *messages* explicites. Enfin, les classes sont organisées en une hiérarchie d'héritage permettant la définition par addition de structure et de comportements.

En particulier, trois notions essentielles sont à la base des langages objets : l'*héritage*, l'*encapsulation* et l'*envoi de message* :

- **L'héritage** : les classes héritent des propriétés et des méthodes (fonctions) de leur superclasses, c'est-à-dire les classes dont elles dérivent, situées plus haut dans la hiérarchie de classe. Elles se spécialisent par addition de nouvelles propriétés (éléments structurels) et de nouvelles méthodes. Eventuellement, certaines méthodes peuvent être *surchargées*, pour particulariser un traitement générale à la hiérarchie, mais spécifique à la classe.
- **L'encapsulation** : les classes encapsulent des données et des fonctions qui leur appartient. Les données sont ainsi protégées et manipulées par du code local. Cet aspect est très important pour une programmation de qualité, c'est-à-dire une clarté et une lisibilité du code amélioré,
- **L'envoi de messages** : un objet communique avec une autre objet par l'intermédiaire d'un message. L'objet récepteur décide seul de la manière de traiter le message qui peut conduire à émettre d'autres messages à d'autres objets, etc. On quitte ainsi la vision centralisée du paradigme procédurale des langages tels que Pascal et C pour lesquels une procédure globale ordonnance l'exécution séquentiel du programme.

Ces trois propriétés sont à la base du succès des langages objets car elles favorisent la *modularité* et la *réutilisation* des programmes. Ces idées ont données lieu à une grande variété de langages. Ainsi, les langages acteurs comme ACT (Lieberman 1987) renforcent l'indépendance entre les objets et la communication par envoi de message, qui sont traités de manière asynchrones, comme des processus. Les systèmes méta-circulaires comme ObjVlisp (Cointe 1987) considèrent les classes comme étant instances de (méta) classes, elles même instances (méta-méta) classes, ce qui permet de modifier la sémantique des classes par l'intermédiaire du langage. Un certain nombre de ces travaux sur l'utilisation de ces langages pour résoudre divers problèmes d'In-

telligence Artificielle ont été menés sur la base de ces travaux, notament par *l'école parisienne* (Pachet 1997) : implémentation efficace de mécanismes d'inférence (Voyer 1989), représentation des hiérarchies de parties et des multi-facettes (Wolinski 1990), représentation des mécanismes d'évaluation (Krief 1990) ou encore connotation et coréférence (Bourgeois 1990).

Enfin, des langages comme LOOPS (Bobrow & Stefik 1988*b*) ou FLAVORS (Moon 1986) traitent des aspects additionnels comme l'héritage multiple, les perspectives et les objets composites.

Les bases de données orientés objets

A la différence des bases de données relationnelles, les Systèmes de Gestion de Bases de Données orientées-objets (SGBDOO) tels que ORION (Kim et al. 1987) ou Versant(Ver n.d.) utilisent comme structure de base les objets et les organisent en une hiérarchie de classes. Ces classes sont définies à l'aide d'un LPOO, (généralement C++). La structure organisationnelle ainsi construite est appelée le *Schéma* de la base de données et les données proprement dites sont les instances des classes définies par le schéma. Le rôle essentiel des SGBDOOs est d'assurer la persistance des données. Bien que relativement puissants, les SGBDOOs n'ont pas réellement réussi à s'imposer dans le domaine. Certainement parce que l'élaboration d'un schéma peut être difficile ou parce que le prix des licences de la plupart des SGBDOOs est souvent relativement élevé. Le standard actuel est encore, depuis les années 70, le modèle relationnel.

1.11.2 Les systèmes de représentation des connaissances à objets

A l'origine, les langages de représentation par objets se sont constitués par tentative d'enrichir le modèle des frames avec de nouvelles idées, en particulier en provenance de la programmation par objets. Les premiers langages, qualifiés de post-frames (Euzenat 1998) préservent l'utilisation de facettes, mais imposent une plus grande discipline dans cette utilisation en limitant la variété initiale. A côté, ils adoptent une distinction générique-spécifique qui se traduit par la division des unités de description, les frames, en *classes* et *instances*. La fonction essentielle d'un SRCO est de stocker, d'organiser les connaissances à l'aide des objets, ainsi que de fournir les méthodes d'inférences nécessaires destiner à compléter l'information disponible au sein du système.

Parmis les Systèmes de Représentation des Connaissances à Objets (SRCO) on pourra citer par exemple les systèmes KRL (Bobrow & Winograd 1977), FRL (Roberts & Goldstein 1977), SHIRKA (Rechenmann 1988, Rechenmann et al. 1990), ROME (Carré 1989), OBJLOG (Dugertil 1988a, Dugertil 1988b), Y3 (Ducournau 1988a) et YAFOOL (Ducournau 1988b) ou encore TROPES (Mariño et al. 1990, SHERPA 1995) devenu TROEPS (Valtchev 1999).

Il est difficile de donner des caractéristiques universelles pour définir un SRCO. Certains systèmes, qualifiés de *hybrides* (Masini et al. 1989, Napoli 1992) permettent l'envoi de messages, l'appel à des fonctions ou d'autres mécanismes propres à des paradigmes aussi variés que la logique avec contraintes, la programmation fonctionnelle ou les langages à objets.

Toutefois, il est possible de donner un aperçu d'un système à objets par contraste avec d'autres formalismes précédemment introduits. Ainsi, les moyens d'exprimer les connaissances, désignés par langages de description, s'articulent autour de la notion d'objets structurés, classe

ou instances.

Représentation des connaissances

Les entités de bases des SRCOs sont les **objets** (ou *instances*), regroupées au sein d'entités plus génériques appelés **classes**. Chaque objet possède une identité qui peut être dans certain cas attribué automatiquement par le système, soit par l'utilisateur. Elle se décrit par un ensemble de couples *attributs-valeurs*, les attributs étant ceux définis par la classe d'appartenance. La valeur d'un attribut peut relever d'un type primitif comme *entier*, *booléen* ou *chaîne de caractères*, ou être instance d'une autre classe. Dans ce dernier cas, l'attribut matérialise une relation entre la classe à laquelle il est attaché, appelé *domaine* de la relation et la classe de l'instance qui constitue sa valeur, appelé *co-domaine* de la relation. Ceci permet de lier les classes entre elles par l'intermédiaire de leur propriétés.

Dans les classes, des *facettes* (ou descripteurs) sont associées aux attributs. Les facettes sont en nombre restreint et servent principalement à préciser le type des valeurs des attributs. Elles servent également à introduire des contraintes ou des mécanismes capables de calculer la valeur d'un attribut (démons) en fonction d'autres valeurs ou encore de gérer les relations entre objets (Euzenat 1999). Ainsi, les facettes de typage permettent de restreindre l'ensemble des valeurs possibles d'un attribut, en précisant les valeurs admissibles par une énumération de valeurs ou par une union d'intervalles, dans le cas des types numériques. Une facette de cardinalité peut permettre de restreindre le nombre maximal d'objets en relation avec l'instance concernée (relation de type $1-n$).

Les classes sont organisées par l'intermédiaire d'une relation d'ordre partiel en une hiérarchie conceptuelle $\mathcal{H} = (\chi, \preccurlyeq, \omega)$ où χ est un ensemble de classes, $\preccurlyeq$ la relation de spécialisation (ou relation d'héritage) et ω est l'élément maximal de χ suivant $\preccurlyeq$ (Simon 2000). Une classe hérite des propriétés de ses ascendants directs, appelés *super-classes*. L'héritage est dit multiple lorsqu'une classe peut avoir plusieurs ascendants.

Dans les SRCO, la sémantique de la relation de spécialisation est celle de l'inclusion ensembliste : une classe s'interprète comme un ensemble d'individus et les individus appartenant à l'interprétation d'une classe doivent appartenir à l'interprétation des super-classes (fig.1.12). De cette interprétation découlent les principes suivants :

- plus une classe est spécifique, plus les domaines de ses attributs sont restreints,
- si une classe est sous-classe de deux classes, les domaines de ses attributs sont forcément inclus dans (ou égaux à) l'intersection des domaines de ces deux classes,
- si un objet est instance d'une classe, il l'est aussi de toutes les super-classes.

Mécanismes de raisonnement dans les SRCO

Les mécanismes d'exploitation des connaissances sont utilisés pour la consultation des connaissances disponibles dans les bases ou bien pour extraire de nouvelles connaissances qui ne sont pas explicitement représentées dans la base.

Au sein d'un système à objets, les mécanismes principaux de raisonnement et de gestion des

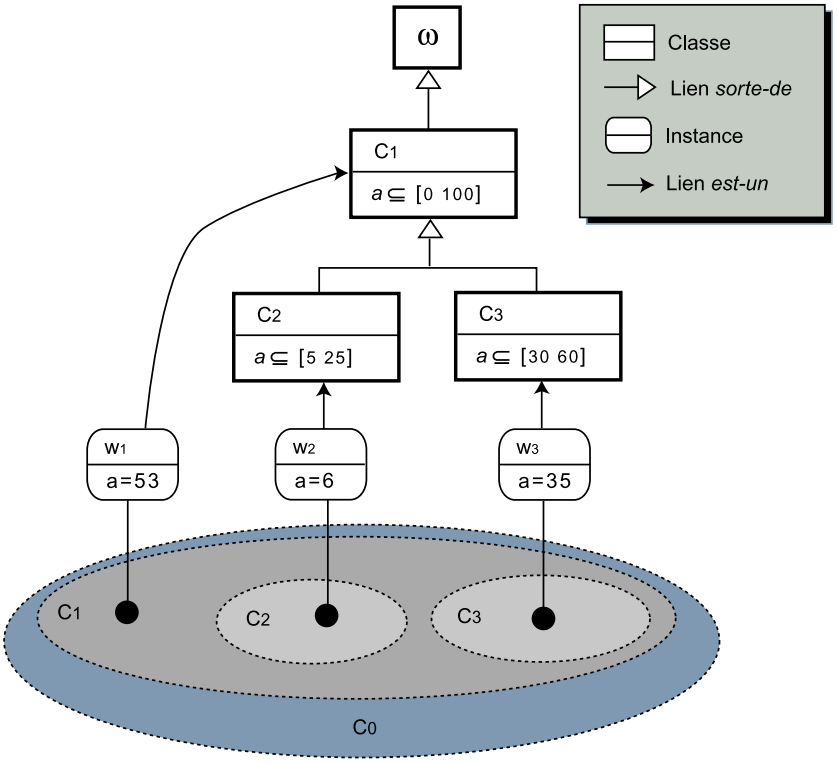


FIG. 1.12 – *Exemple d'une représentation à objet à héritage simple. Les classes sont organisées en une hiérarchies de spécialisation. Les instances doivent respecter les contraintes définies par le domaine de définition des attributs. L'extension d'une classe plus spécifique est incluse dans l'extension de la classe la plus générale.*

contraintes d'intégrités sont :

- le **partage de propriétés**. Le mécanisme d'héritage met en jeu de façon *dynamique* ou *statique* le partage de propriétés entre classes. En héritage statique, les propriétés sont recopiées explicitement dans les sous-classes. Lorsque l'héritage est dynamique, les propriétés ne sont pas recopiées dans les sous-classes : le mécanisme d'héritage effectue alors un parcours du graphe de classe pour récupérer les propriétés des super-classes. Dans les deux cas, une classe hérite de toutes les propriétés de ses super-classe et peut éventuellement les surcharger.
- le **calcul de la valeur** d'un attribut. Des facettes d'inférences peuvent être associées à un attribut et permettent de calculer automatiquement une valeur. La plus utilisées est la facette *par défaut* qui permet de donner une valeur à priori à un attribut en l'absence de tout autre valeur renseignée explicitement. Une facette particulière, l'attachement procédural, permet d'effectuer un calcul et d'associer le résultat du calcul à l'attribut. Plusieurs facettes peuvent être attachées simultanément à un même attribut, ce qui implique qu'une stratégie soit mise en place pour décider de l'ordre d'application de ces facettes (Napoli 1992),
- la **classification** concerne les instances ou les classes. La classification de classes consiste à placer une nouvelle classe dans le graphe des classes en recherchant ses super-classes les plus spécifiques et ses sous-classes les plus générales. La classification d'instances consiste à

trouver la classe d'appartenance la plus adaptée d'un individu dans la hiérarchie de classe (Napoli 1992, Ducournau et al. 1999).

– le **filtrage**. Pour la recherche d'un ensemble d'instances qui possèdent certaines propriétés, le mécanisme de filtrage permet de regrouper les individus s'appariant à un *filtre*. Ce mécansime s'apparente aux requêtes des bases de données.

1.11.3 Les objets composites

Le modèle de représentation des connaissances que nous présentons au chapitre suivant étant basé sur la notion d'*objets composites*, nous développons les différentes interprétations de le relation de composition utilisés par les systèmes centré-objets qui manipulent les *objets composites* (Napoli 1992, Drews 1993).

La relation de spécialisation (ou subsomption) est fondamentale dans le modèle objet au regard du grand nombre de mécanismes basés sur cette relation : inférence par héritage, classification, recherche des Ascendants les Plus Spécifiques (APS), etc. Cependant, certains systèmes à objets, tels que LOOPS (Bobrow & Stefik 1988*b*, Masini et al. 1989) ou SRL (Fox et al. 1986) par exemple accorde un rôle particulier à la relation de composition et aux objets composites.

Un objet composite est un objet complexe pour lequel est défini un ensemble de sous-parties. Chaque sous-partie est elle-même décrite par un composant. Un objets composite est relié à chaque sous-partie par un relation de type « partie-de » qui peut être interprétée de différentes façons (Napoli 1992, Drews 1993). Winston et al. (1987) ont identifié 6 catégories de relations de compositions (cf. tableau 1.1), qui se distinguent par trois aspects :

– **Fonctionnalité** : la relation entre le tout et les parties est-elle fonctionnelle ? C'est le cas lorsqu'une partie réalise une sous-tâche de la tâche réalisée par le tout.
– **Homogénéité** : les parties sont-elles du même type que le tout ?
– **Séparabilité** : les parties sont-elles séparables du tout ?

Relation	Exemple	Fonctionalité	Homogénéité	Séparabilité
Composant / tout	Moteur-voiture	+	-	+
Membre / collection	Arbre-forêt	-	-	+
Portion / masse	Grain-sel	-	+	+
Matière / objet	Bois-armoire	-	-	-
Trait / activité	Payer-faire des courses	+	-	-
Lieu / Aire	Oasis-Désert	-	+	-

TAB. 1.1 – *Les 6 types de relations de compositions, d'après (Winston et al. 1987)*

La relation de composition est une relation transitive, non réflexive et antisymétrique :

– **non réflexive** car un objet n'est pas une partie de lui-même,
– **asymétrique** car si a est une partie de b, alors b n'est pas partie de a,
– **transitive**, car si a est une partie de b et b une partie de c, alors a est une partie de c.

Cette dernière propriété de transitivité n'est pas toujours valide. Par exemple, on peut dire (Winston et al. 1987) :

`Le carburateur fait partie du moteur` (relation composant–objet entier)

`Le moteur fait partie de la voiture` (relation composant–objet entier)

`Alors, le carburateur fait partie de la voiture` (relation composant–objet entier)

Par contre, on ne peut pas dire :

`Le bras de Paul fait partie-de Paul` (relation composant–objet entier)

`Paul fait partie du département de philosophie` (relation membre-collection)

En général, la propriété de transitivité est garantie lorsque la relation "partie-de" est entendue avec un seul des sens mentionné ci-dessus. Dans les cas où différentes sémantiques interviennent, la transitivité de la relation peut-être remise en question.

La plupart des systèmes de représentation à objets manipulant des objets composites ne font pas la distinction explicite entre ces différents types de relation de composition. Ces systèmes peuvent se classer en trois groupes selon le type de définition adoptée (Drews 1993) :

- Définition structurelle des objets composites : un tout et des composants qui sont des objets à part entière tel que dans LOOPS (Bobrow & Stefik 1988*b*, Masini et al. 1989),
- La sémantique de la relation est donnée explicitement par le concepteur de la base, tel que dans les systèmes de représentation des connaissances tels que dans les langages Objlog (Dugertil 1991) ou SRL (Fox et al. 1986),
- Définition fonctionnelle : la relation de composition est le vecteur du maintien de la cohérence entre les sous-parties et le tout par propagation de contraintes entre objets. (Borning 1981, Sussman & Steele 1980, Davis 1987).

1.11.4 Évolution et points de vue dans les SRCOs

Certains modèles à objets mettent en avant l'évolution des connaissances et la possibilité d'exprimer plusieurs points de vue sur les connaissances, comme par exemple le système de gestion de bases de connaissances à objets TROEPS (Mariño et al. 1990, SHERPA 1995, Valtchev 1999). TROEPS (originellement TROPES) est le successeur du système SHIRKA (Rechenmann et al. 1990), développé par l'équipe SHERPA de l'INRIA Rhône-Alpes. La conception de TROEPS a été motivée par le besoin d'apporter une solution à un ensemble de problèmes existant dans les SRCO (Carré & Comyn 1988, Carré 1989, Napoli 1992, Drews 1993, Euzenat 1999, Valtchev 1999). D'une part, comment gérer d'une manière efficace l'évolution des connaissances et d'autre part comment autoriser des classifications multiples sein différents *points de vue* sur ces connaissances. La figure 1.13 illustre la vision multi-points de vue, telle qu'elle est considérée dans le système TROEPS. La notion de *passerelle* permet de lier les différents points de vue, de manière à permettre d'exprimer des relations d'inclusion entre classes de points de vue différents.

Pratiquement, la logique des points de vue est très semblable à la logique de composition. Cependant, elle ne possède pas une sémantique aussi riche. En effet, lors d'une classification, l'absence avérée d'une sous-partie sera prise en compte comme une information, alors que l'absence

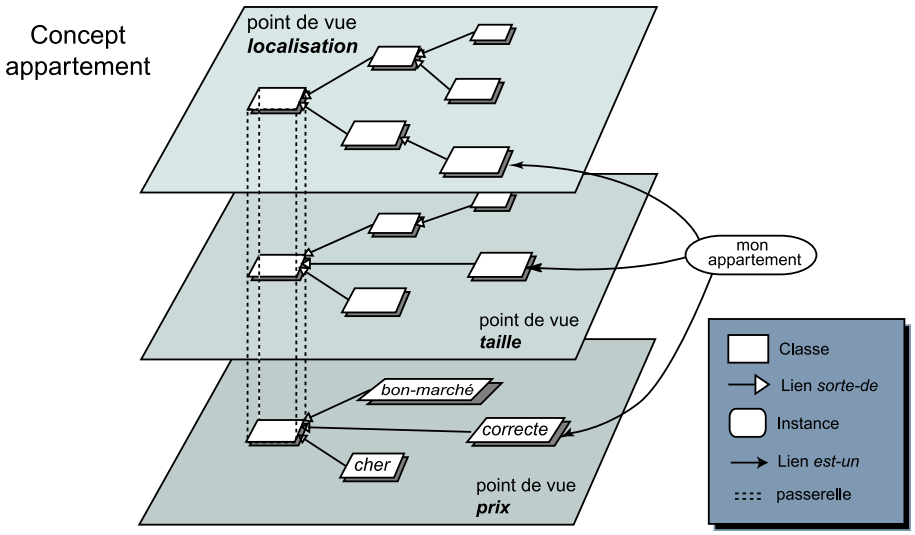


FIG. 1.13 – *Exemple d'une base de connaissances multi-points de vue en TROEPS (Valtchev 1999). Le concept* ***appartement*** *est représenté selon trois points de vue :* ***prix****,* ***taille*** *et* ***localisation****. Une instance est visualisée avec ses liens d'attachement sous chaque point de vue.*

d'un point de vue n'a aucune signification classificatoire (Euzenat 1999).

1.12 Conclusion : quel modèle pour la systématique ?

1.12.1 Avantages des représentations à objets

Le modèle objets apparait comme un moyen particulièrement adapté à la représentation de connaissances complexes sur un domaine d'étude en cours d'élaboration, comme peuvent l'être certaines branches de la systématique. Ce qui est le cas dans le domaine des coraux.

Les entités de représentation de base sont les objets, qui dénotent des entités individuelles, et les classes qui expriment les différentes catégories du domaine. Le modèle des objets composites en particulier s'adapte bien à la structuration des descriptions d'objets biologiques. Cette structuration a été souligné dans Conruyt (1994), comme le moyen privilégié d'« améliorer la robustesse des systèmes d'aide à la description, à la classification et à la détermination des objets biologiques ».

Une approche naturelle pour représenter la taxonomie des espèces avec le modèle objet, consiste à modéliser les classes de manière à ce que chacune d'entre elle dénote un nom de taxon particulier. En effet, les classes sont structurées en hiérarchies (parfois multiples) et la sémantique associée à la subsomption correspond à celle de l'inclusion ensembliste, comme dans la taxonomie. Dans cette approche, les classes modélisent des taxons, les attributs correspondent aux caractères et les objets, aux représentants des taxons, c'est-à-dire les individus.

De plus, le modèle des « points de vue » nous semble d'un grand avantage pour la systématique. En effet, dans ce domaine il est non seulement nécessaire de classer les spécimens selon

une taxonomie prédéfinie[30] , mais également d'assurer un suivi et une gestion de collection. Il est donc nécessaire de disposer parallèlement de plusieurs *systèmes classificatoires* afin de réaliser ces multiples objectifs (gestion de collection, classification, identification, etc.).

Cependant, malgré ses avantages, nous pensons que le modèle objet en représentation des connaissances, tel qu'il est communément admis par les informaticiens n'est pas totalement adapté à la systématique.

1.12.2 Limites du modèle objet classique pour la systématique

Alors qu'il existe une standardisation de la *Taxonomie des Espèces* au niveau mondial, les biologistes utilisent fréquemment de nombreuses taxonomies alternatives, telles que les clés d'identification(Pankhurst 1991, Lebbe 1998, Vignes 2000), ou bien des taxonomies locales destinées à être intégrées ultérieurement aux taxonomies standarts (Beach et al. 1993). Pourtant, toutes ces taxonomies concernent bien le même ensemble d'individus. Comme le souligne (Euzenat 1999), la multiplicité des taxonomies biologiques conduit rapidement à adopter une attitude relativiste vis-à-vis de ces classifications, qu'il est difficile d'accorder à la volonté de construire des « ontologies » (Guarino 1998). Par ailleurs, au niveau de la définition même des taxons, il y a de nombreuses différences d'interprétations qui portent sur la structuration des caractères descriptifs attachés aux taxons. Il n'existe généralement pas de consensus à l'heure actuel quant à l'organisation de ces caractères, de leurs dépendances respectives et de la terminologie utilisée pour les exprimer.

Certes, le modèle des points de vue apporte une réponse forte pour la conception de taxonomies multiples, qui caractérisent différentes interprétations de la taxonomie des Espèces. Nous pensons qu'il faut aller plus loin et introduire la notion de point de vue non seulement dans le processus de construction des taxonomies, mais également au niveau de la structuration ds caractères descriptifs caractérisants les taxons.

Le modèle que nous proposons introduit la notion de points de vue à deux niveaux différents. D'une part, la taxonomie est scindée en deux niveaux, les taxons de haut niveau (Ordre, famille) dont la structure est généralement consensuelle, et les taxonomies de plus bas niveau (de la famille à l'espèce) correspondant à une interprétation « plus personnelle » de l'expert. Dans le domaine des coraux, les experts ne s'accordent pas sur le nombre d'espèces et sur le statut de certains *écomorphes*.

Variabilité et imprécision

La construction d'une taxonomie de classes se base généralement sur la donnée d'un *critère d'homogénéité* des groupes d'objets (Valtchev 1999). En biologie, il est fréquent que la variabilité *intra-spécifique* soit plus grande que la variabilité *inter-spécifique*, c'est-à-dire que la distance entre deux individus d'une même classe soit plus grande que la distance entre deux individus de classes différentes. C'est particulièrement vrai pour les coraux, même au niveau de la colonie. La variabilité *intra-coloniale* est parfois plus importante que la variabilité *intra-spécifique* ! D'autre part, le modèle objet classique ne permet pas de différencier la variation (présence simultanée)

30. sujette à controverse et souvent remise en cause

de l'imprécision (choix difficile entre deux valeurs) dans les descriptions. Notre modèle donne la possibilité d'effectuer cette distinction dans les descriptions.

Exception

Chaque spécimen biologique, qu'il soit considéré comme central ou marginal, porte sa propre originalité et fait ainsi partie intégrante de la couverture de la classe. A ce titre, les cas exceptionnels ont autant d'importance que les cas typiques, voir même plus d'importance pour certains biologistes[31], c'est pourquoi les biologistes affirment que dans la nature, l'exception est la règle !

Or, dans le modèle objet, la définition de la relation de spécialisation interdit l'introduction des exceptions dans les domaines des classes (c'est-à-dire déroger à la règle de spécialisation des domaines des attributs). Comme le souligne (Brachman & Schmolze 1985*b*), introduire des exceptions dans le domaine des classes permettrait inévitablement de définir « un éléphant comme un singe qui ne grimpe pas aux arbres ». De même, il n'est pas possible pour un objet de prendre un état non prévue par la classe à laquelle il est attaché.

Comment donc autoriser l'exception ? Plus précisément, comment permettre des états exceptionnels non prévu initialement par le domaine des attributs des classes ? Le modèle objet semble n'apporter que peu d'éléments de solutions sur ce point, tout au moins, sans modifier la définition des classes.

Héritage multiple et multi-instanciation

Le modèle objet autorise la construction des hiérarchies de classes multiples[32], la définition de *classes virtuelles* (Simon 2000), ainsi que la *multi-instanciation*, c'est-à-dire la possibilité pour une instance d'appartenir à plusieurs classes.

La taxonomie des êtres vivants est une hiérarchie simple. Créer des hiérarchies multiples n'est donc pas une nécessité. D'autre part, un spécimen ne peut appartenir simultanément à plusieurs taxons. Le cas échéant, c'est une imprécision et non une propriété intrinsèque.

Propriétés nécessaires et contingentes

Enfin, le modèle objet ne permet pas de distinguer entre des propriétés (attributs) *nécessaires* et des propriétés *contingentes* dans la définition des classes. Toute propriété est nécessaire, c'est-à-dire qu'un objet doit nécessairement posséder les propriétés définies par la classe pour pouvoir y être attaché.

Or, les monographies sont riches de qualificatifs comme "la plupart", "fréquemment", "parfois", etc. qu'il est difficile de prendre en compte sans introduire la possibilité d'exprimer des propriétés contingentes, c'est-à-dire des propriétés parfois absentes des descriptions, qui ne remettent pas en cause l'appartenance de l'instance à la classe.

31. Gérard Faure, communication orale.
32. aussi nommé multi-spécialisation ou multi-généralisation (Euzenat 1999).

Deuxième partie

Le modèle CoDesc

Chapitre 2

CoDesc, un modèle de représentation des connaissances descriptives et classificatoires

Sommaire

2.1 Introduction

Nous proposons dans ce chapitre un modèle de représentation hiérarchique des connaissances qui permet d'exprimer des concepts, leurs inter-relations et de décrire des individus, exemples de ces concepts. Ce modèle est baptisé CoDesc pour représentation des connaissances **descriptives** et **classificatoires**. Il est particulièrement adapté à la modélisation de connaissances complexes, issues d'observations de la nature, sous la forme de représentations structurées.

2.1.1 Genèse du modèle

Les origines de ce modèle sont « les logiques descriptives en sciences de la vie » élaborées par Noël Conruyt (1994) pour faciliter la conception de bases de connaissances robustes, à travers l'acquisition de descriptions d'objets biologiques par des experts. Ces logiques descriptives ont initialement été formalisées à l'aide des objets symboliques (cf. § 1.9.5) et programmées au sein d'un système appelé *HyperQuest* (Conruyt 1994), dans le langage HyperCard. Ce système s'est montré d'une grande aide, en particulier pour l'acquisition des connaissances des systématiciens, grands explorateurs et observateurs de la nature et de son organisation. Une application sur les éponges marines appartenant au genre *Hyalonema* (Renard et al. 1996) a été validée par les experts du domaine.

Cependant, le système proposé a montré certaines limites, en particulier la difficulté de pouvoir aisément faire évoluer les bases de connaissances et l'impossibilité d'implanter des méthodes d'analyse efficaces (pour la détermination et la classification automatique) du fait de la faiblesse du langage de programmation hôte.

Dans le cadre de cette thèse, le modèle des *logiques descriptives* a évolué vers un modèle déclaratif orienté-objets de représentation des connaissances descriptives, à base d'objets composites. Il intègre en particulier les notions de hiérarchies de concepts (§ 2.6) et de valeurs complexes (cf.§ 2.9). Il a été implémenté au sein de la plate-forme $\mathcal{IKBS}$, à l'aide de classes du langage orienté-objet Java (cf. chap :9).

2.1.2 Objectif

L'objectif de ce modèle de représentation est de permettre à un expert, ou un groupe d'experts, d'expliciter et d'organiser toute les connaissances descriptives utiles, relatives à un domaine donné, en différents concepts. Ces concepts représentent des catégories de connaissances. Dans notre domaine d'application, les catégories sont connues par avance et ne sont que très rarement remises en cause par les experts. Par contre, l'expression des différents concepts est souvent

sujette à caution : la terminologie évolue, les descriptions sont révisées fréquemment, les commentaires critiqués et les attributs modifiés. La connaissance évolue donc, mais essentiellement dans une dimension horizontale (la *composition*) [33], plutôt que dans une dimension verticale (la spécialisation) : l'ordre de la nature étant souvent le fruit d'un long travail (parfois de plusieurs siècles, depuis Linnaeus [34]) et donc très difficile à remettre en cause.

Le modèle actuel est utilisé et testé depuis plusieurs années par un groupe d'experts en systématique des *Scléractiniaires* pour la conception d'une *Base de Connaissances sur les Coraux de l'Archipel des Mascareignes* [35]. (Faure et al. 1999, Conruyt et al. 1998).

2.1.3 Originalité du modèle

CoDesc diffère cependant des représentations à objets classiques et des autres types de langages de représentation hiérarchiques des connaissances. En particulier, la plupart de ces systèmes adoptent une **approche descendante** lors de la construction d'une taxonomie de classes : les classes sont formées par spécialisations successives, par ajout de nouveaux descripteurs dans les sous-classes. Le but essentiellement recherché est une **caractérisation intensionnelle** des classes, c'est-à-dire un ensemble de **conditions nécessaires** d'appartenance d'une instance à la classe.

Dans le modèle CoDesc, les concepts sont décrits par une entité de plus haut niveau que la classe, appelée **modèle descriptif** qui représente l'ensemble des *connaissances de fond* du domaine modélisé par l'expert. Il peut être interprété comme le plan d'organisation des individus appartenant au concept et contient l'ensemble des caractères *observables* utiles pour la description des individus. Un aspect essentiel du modèle est de distinguer deux types de descripteurs exprimants des conditions d'appartenance des individus au concept : celles qui peuvent être absentes des descriptions sont dites *contingentes*, [36] car elles ne peuvent remettre en cause l'appartenance de l'individu au concept. Les autres, qualifiés de *nécessaires* [37] forment la partie intensionnelle du concept et doivent obligatoirement (conditions nécessaires) être renseignées dans les descriptions. Cette particularité du modèle offre l'avantage majeur d'introduire une logique modale (est ou n'est pas, parfois présente) pour représenter des concepts dans les domaines de connaissances liées à l'observation du monde réel, pour lesquels une logique booléenne est rarement satisfaisante.

Le modèle offre un cadre d'expression très riche pour la description des concepts à l'aide d'un ensemble de **descripteurs** [38] de différents types : schémas, composants et attributs, auxquels

33. la composition des objets caractérise une dimension horizontale de la représentation des connaissances et la *spécialisation* des objets caractérise une dimension verticale.

34. Linnaeus (1707-1778) est le fondateur de la *nomenclature binominale*. Cette nomenclature à permis à des générations de scientifiques d'améliorer la classification des organismes et d'identifier les spécimens sur la base des organisations taxonomiques.

35. situé dans le sud-ouest de l'Océan Indien, près de Madagascar, comprenant en particulier les îles de La Réunion, Maurice et Rodrigues

36. **Contingent,e** adj. Qui peut se produire ou non, être ou ne pas être (par opposition à nécessaire). *Le Petit Larousse, éd. 1998*

37. **Nécessaire** adj. (lat. *necessarius*) Dont on a absolument besoin ; essentiel, primordial. Qui ne peut pas ne pas se produire dans des conditions données, au sein d'un processus donné (par opposition à *contingent.*) *Le Petit Larousse, éd. 1998*

38. on regroupe sous le terme générique **descripteurs** l'ensemble des éléments utilisés pour la description des individus. Notons que l'on utilisera également parfois le terme descripteur pour désigner la personne qui décrit.

sont attachés des relations, des règles et des données contextuelles. Les individus sont décrits à l'aide de *valeurs complexes* (multiples, taxonomiques, conjonctives, disjonctives, inconnue, etc.), nécessaires pour la description des individus relevant du concept.

De plus, la sémantique originale associée à la spécialisation et à la généralisation des concepts et des composants offre une assistance aux experts pour construire une Taxonomie par une **approche ascendante** : les descriptions d'individus favorisent la mise au point de modèles descriptifs spécifiques par un raisonnement *inductif*, les propriétés nécessaires de ceux-ci étant généralisées dans les modèles de plus haut niveau (§ 2.6). Il est en effet très difficile d'élaborer *a priori* des modèles très généraux, sans l'aide d'un ensemble de descriptions fines et détaillées d'individus (l'Observé), qui expriment la variabilité et la richesse des objets de la Nature.

Des *points de vue* peuvent être exprimés pour représenter différents aspects d'un même concept ou bien pour permettre aux experts de travailler de concert pour la conception de bases de connaissances traitant du même domaine. La notion de point de vue se retrouve également exprimée, d'une manière quelque peu différente de celle observée dans les systèmes centrés-objets, comme l'avons illustré au chapitre précédent.

2.2 Bases de connaissances en CoDesc

Le modèle CoDesc est utile pour construire des bases de connaissances **descriptives** et **classificatoires**, à travers une modélisation des concepts du domaine et l'acquisition d'un ensemble de descriptions des individus relatifs au domaine. Par descriptives, nous comprenons les connaissances qui portent sur la définition des concepts et la description des individus (l'extension) relevant des concepts ainsi représentés. Par classificatoires, nous entendons toute connaissance qui portent sur la classification des individus, la manière d'établir des classifications et de classifier les individus.

Une base de connaissances comprend l'ensemble des connaissances modélisées par un expert (ou un groupe d'experts) et constitue ainsi un modèle du domaine dans un contexte particulier.

Trois entités principales définissent une base de connaissances : les **concepts**, les **cas** et les **informations contextuelles**[39].

Définition 2.1 *Une base de connaissances $\mathcal{BC}$ est définie par le triplet $\{\mathcal{C},\Omega,\mathcal{K}\}$, où :*

- $\mathcal{C} = \{\mathcal{C}_1,\ldots,\mathcal{C}_r\}$ *désigne un ensemble de concepts.*
- $\Omega = \{\omega_1,\ldots,\omega_k\}$*, un ensemble de cas, attachés à l'extension des concepts de* $\mathcal{C}$*.*
- $\mathcal{K} = \{k_1,\ldots,k_l\}$*, un ensemble de références à des connaissances non formalisées, extérieures à la base de connaissances. Elles sont désignées par le terme générique données contextuelles.*

39. ou *données contextuelles*.

Terminologie

Par la suite, nous utiliserons par la suite plusieurs termes différents pour désigner les « objets » de l'étude, sachant que nous entendons :

- le terme *individu* désigne l'objet en temps qu'entité réelle, indépendamment de sa classe (ou concept) ;
- *instance* désigne un objet en tant qu'un élément particulier de l'extension d'une classe ou d'un concept donné ;
- *cas* dénote une donnée informatique qui représente l'objet ;
- et enfin *description* correspond au produit de l'analyse de l'objet effectué par un observateur ou d'un expert lorsqu'il s'attache à décrire l'objet.

Ce dont nous souhaitons nous intéresser par la suite, c'est essentiellement l'objet qui a été décrit, informatisé et attaché à un concept. Nous utiliserons donc de préférence le terme *instance*.

2.3 Concept et modèle descriptif

Un **concept** est le représentant d'un groupe d'individus. Il possède une description modélisée par une structure de données arborescente, appelée **modèle descriptif**.

Nous utiliserons par la suite les termes de *concept* ou de *modèle descriptif*, de manière équivalente, sachant cependant que un concept dénote une *abstraction intellectuelle* (cf. chap. 1) alors qu'un *modèle descriptif* correspond à la description d'un concept, une représentation informatique élaborée dans un certain contexte.

La figure 2.1 illustre un exemple d'une partie d'un modèle descriptif. Nous exposons par la suite, la nature des différents éléments qui le constituent.

Le modèle descriptif regroupe l'ensemble des caractéristiques utiles pour la description des individus appartenant au concept.

Certaines caractéristiques sont qualifiées de *nécessaires*, auquel cas une instance doit posséder obligatoirement ces caractéristiques pour appartenir au concept. Les autres ont uniquement un rôle utile à la description et sont utilisées pour affiner la description des individus ; Elles sont qualifiées de *contingentes* car peuvent être absentes des descriptions sans remettre en cause l'appartenance de l'individu au concept. Cette distinction est fondamentale dans le modèle proposé, car elle permet de considérer les concepts selon deux points de vue :

- **Point de vue intensionnel**. La restriction du modèle descriptif dénotant un concept à l'ensemble des *caractéristiques nécessaires* permet une caractérisation de l'*intension généralisée* [40] du concept. Cette restriction est appelée **sous-modèle intensionnel**. Toute instance doit obligatoirement posséder ces caractéristiques, c'est-à-dire qu'elles doivent être renseignées dans la description, pour appartenir au concept.

40. L'**intension généralisée** donne des conditions nécessaires d'appartenance d'un individu à une classe ou un concept (cf. chap. 1.4.1), par opposition à l'**intension stricte** qui exprime des conditions *nécessaires et suffisantes*.

– **Point de vue extensionnel**. Un modèle descriptif possède la totalité des *descripteurs observables* pour décrire les individus. Il offre ainsi une vision synthétique complète de l'univers possible de description, appelé l'**Observable**.

Chaque individu appartient à un unique concept $\mathcal{C}$, et par extension, à l'ensemble des concepts qui subsument $\mathcal{C}$. Par analogie avec les représentations à objets, il est dit *instance* du concept. Nous disons qu'un individu appartient à l'**extension** de son concept, qui est l'ensemble des individus de l'univers du discours pouvant être recouverts par ce concept. La description d'un individu[41] dépend étroitement de la structure du modèle descriptif dénotant le concept dont il est instance. Ainsi, un concept est muni de prérogatives *ontologiques* sur l'ensemble de ses instances car il définit leur structure et leur identité. Par ontologique, nous entendons ici ce qui est lié à l'essence même de la notion représentée. Un concept définit donc une structure commune pour toutes les instances qu'il recouvre, ce qui permet par exemple de comparer les instances sur la base de cet ensemble de descripteurs communs.

2.3.1 Expression d'un concept par un modèle descriptif

Un modèle descriptif (fig. 2.1) contient l'ensemble des *connaissances de fond* d'un sous-ensemble de connaissances du domaine modélisé par l'expert. D'un point de vue informatique, c'est une structure de donnée complexe, élaborée pour représenter un concept qui désigne une représentation abstraite, structurée sous la forme d'un arbre, d'un domaine particulier. On peut considérer le modèle descriptif comme le plan d'organisation des descriptions portant sur l'ensemble des individus (les cas) couverts par le domaine (Conruyt 1994).

En particulier, pour la représentation de connaissances en systématique, un modèle descriptif désigne l'ensemble de la connaissance d'un groupe taxonomique particulier (Ordre, Famille, Genre ou Espèce). Le choix de la granularité imposée à un modèle descriptif particulier relève d'un choix de conception lié à la complexité du taxon considéré. Par exemple, dans la systématique des coraux, les plus petites familles sont constituées de seulement de quelques espèces (les *Astrocoeniidae* par exemple), alors que d'autres sont constituées de plusieurs genres et de quelques dizaines d'espèces (*Pocilloporidae*). Pour ces « familles nombreuses », le concepteur de la base aurait pu choisir une granularité du modèle descriptif au niveau du genre plutôt qu'au niveau de la famille, afin de diminuer la complexité et le nombre de descripteurs nécessaires.

La figure 2.1 illustre un modèle descriptif dénotant le concept de la famille des *Pocilloporidae*[42]. Les éléments qui composent le modèle sont : le schéma (1), les composants de type "points de vue" (2), les composants réels (3), les attributs (4) et les relations de composition (5). (6) est l'attribut de classe **Taxon** (cf. 2.7). (7) sont des composants « absents possibles » qui caractérisent les racines de sous-arbres possiblement absents des descriptions (et également de certains sous-concepts). A certains descripteurs, sont attachées des règles, des annotations, ou des données contextuelles qui n'apparaissent pas explicitement sur le schéma global du modèle. La présence des données contextuelles est cependant symbolisée par un nombre (voir par exemple l'attribut localisation de l'objet contexte) correspondant au nombre d'occurrences des relations de référence vers les données contextuelles.

41. appelé indifféremment cas, objet ou instance dans notre modèle
42. Une des 16 familles de *Scléractiniaires*.

D'une certaine manière, un modèle descriptif peut être interprété comme un *représentant maximal* du concept, correspondant à la couverture totale de l'extension du concept. Il est cependant une entité abstraite, c'est-à-dire qu'il ne correspond pas à une entité observée.

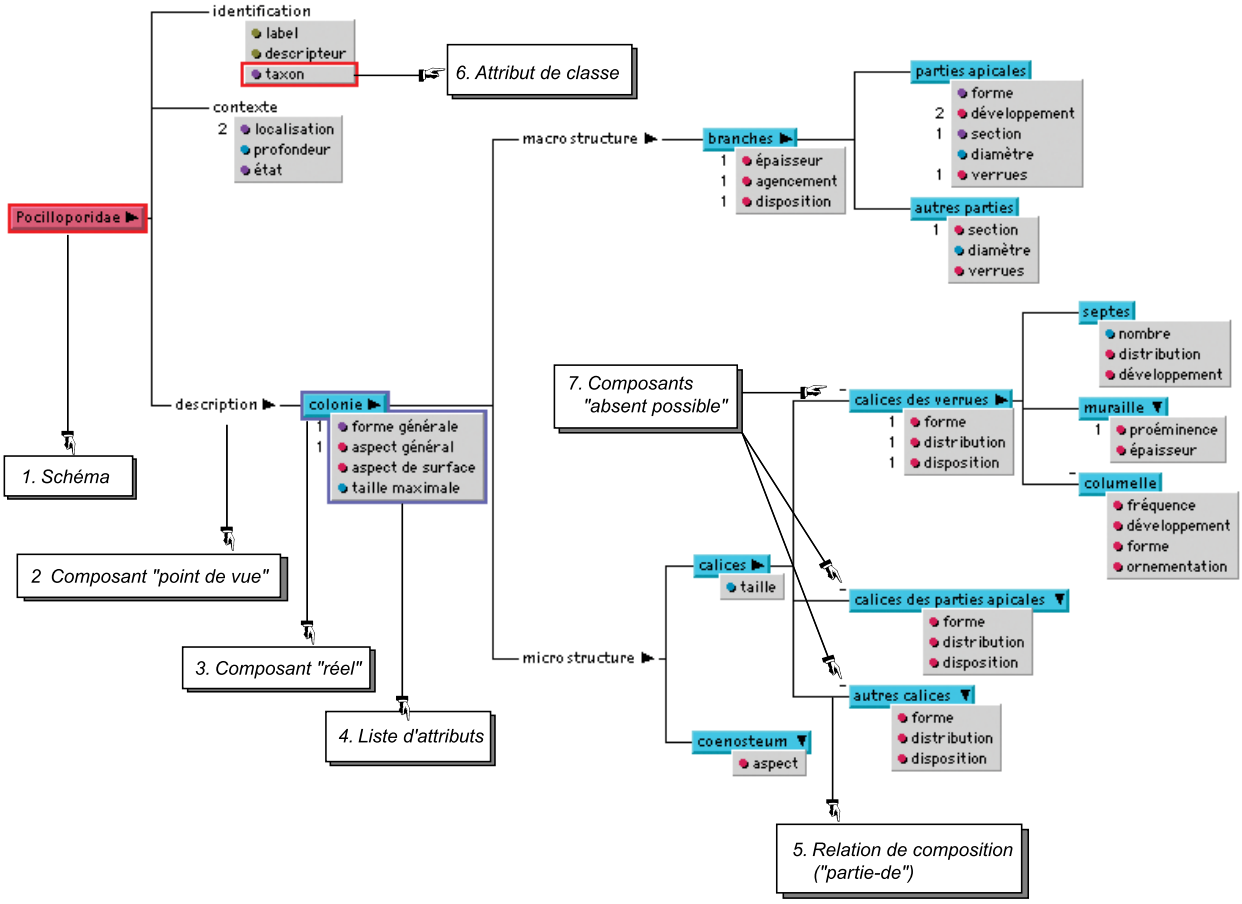


FIG. 2.1 – *Le **modèle descriptif** de la famille des **Pocilloporidae**. Extrait de la bases de connaissances sur les coraux des Mascareignes. Experts : Gérard Faure et Michel Pichon.*

2.3.2 Les composants

Un modèle descriptif est formé par l'agrégation d'un ensemble d'objets, appelés *composants* [43]. Un composant particulier, appelé **schéma**, est le représentant du concept. Les autres composants sont reliés au schéma ou entre eux par la relation *partie-de*, qui est orientée, et décrivent chacun une partie du modèle. La relation inverse, *composé-de*, relie les composants au schéma ou à d'autres composant. Un composant ne peut pas être relié à lui même, ni à un autre composant avec lequel il est déjà en relation. La structure induite par la relation de composition est une *arborescence*, dont l'unique composant de type *schéma* est la racine du modèle descriptif.

Les composants sont très semblables aux *objets composites* des représentations à objets. Comme nous l'avons mentionné, la représentation des objets composites offre des interprétations

43. les composants sont aussi appelés **objets composites** dans des représentations à objets. Par la suite, nous emploierons indifféremment les deux terminologies

différentes selon le sens accordé à la relation de composition (voir § 1.11.3). Même si la sémantique des relations entre composants ne peut-être définie par l'expert, les relations prennent des sens différents selon la nature des composants reliés.

En effet, quatre variables booléennes permettent de préciser la sémantique associée aux composants dans CoDesc. Ces variables sont appelées *propriétés internes* (ou propriété intrinsèques) :

- la **fictivité** caractérise un composant de type *point de vue*,
- l'**absence possible** autorise le composant a ne pas être instancié dans les descriptions,
- la **multiplicité** offre la possibilité d'instancier plusieurs fois le composant.
- la **spécialisation** permet de construire une hiérarchie locale de spécialisation du composant.

Ces différents statuts font partie de la définition des composants. Par défaut, un composant ne possède aucune de ces propriétés, il est dit *observable*.

Les *propriétés externes* caractérisent les relations du composant vers d'autres entités du modèles : des attributs (qui le renseignent et affinent sa description), des sous-composants (sous-parties), des règles ou bien des références vers des données contextuelles. Ces propriétés externes (relations) sont définies dans la définition du composant, mais la définition précise des entités reliées est extérieure à celle du composant.

2.3.3 Composant observable

Un composant *observable* dénote un objet réel du domaine considéré. Il est dit observable car il peut être décrit et renseigné par un observateur. [44]. Par défaut, un composant nouvellement créé est observable et son statut peut être modifié *a posteriori*.

2.3.4 Point de vue

Un composant peut également correspondre à un objet fictif, abstrait et il est alors assimilé à un *point de vue*. La propriété booléenne **fictivité** du composant est alors *vrai*.

Les points de vue sont très utiles pour représenter différents aspects d'un même concept. Par exemple, pour la description d'objets biologiques, on pourra définir un point de vue *description* sur les caractéristiques morphologiques, un point de vue *écologie* sur le rôle de l'individu dans son écosystème ou encore un point de vue *gestion de collection*.

Chaque point de vue regroupe un ensemble de descripteurs et de sous-composants permettant d'affiner la définition du point de vue. Un point de vue est la racine (et donc potentiellement le schéma) d'un sous-modèle descriptif caractérisant un point de vue particulier sur le concept. La figure 2.2 illustre différentes perspectives introduites par les points de vue P_1, P_2, P_3 et P_4 dans le concept $\mathcal{C}$. Les points de vue définissent en quelques sortes différents *axes* de décomposition dans le plan de composition, qui particularisent le point de vue dans une dimension donnée. Cette dimension est symbolisée par un *axe de point de vue*. Un point de vue peut lui-même être particularisé en plusieurs sous-points de vue (fig. 2.2).

44. Il est parfois nécessaire d'utiliser des instruments appropriés : loupe binoculaire, microscope, ou bien des procédés d'analyses (biochimique, génétique, cytologique) pour le décrire.

Dans l'exemple de la fig. 2.1, le modèle descriptif est constitué de trois points de vue : *identification*, *contexte* et *description* qui définissent trois axes différents de décomposition. Le contexte va s'attacher à décomposer le spécimen corallien selon tout ce qui a trait au contexte de la description : l'endroit où l'échantillon a été découvert, la profondeur de récolte, ou encore l'état dans lequel il se trouve (bon état, fragment de colonie, détérioré, etc.). L'axe du point de vue *description* d'autre part va constituer la racine d'une décomposition morphologique de l'échantillon, à l'aide de deux *sous points de vue*, la *macro-structure* (observation à l'oeil nu) et la *micro-structure* (observation au microscope).

D'une manière intuitive, on pourrait associer un angle entre deux axes, dont la valeur correspond à une distance séparant les points de vue. Ainsi, dans la figure 2.2, l'angle α entre P_2 et P_4 est plus grand que β entre P_2 et P_3, ce qui est conforme à l'intuition, étant donné que P_3 est un sous point de vue de P_2. D'une manière générale, un angle de 90° correspond à deux axes orthogonaux, sans aucun rapport entre eux au sens de la décomposition. Un angle de 180° a deux dimensions diamétralement opposées. Pour un angle de 0°, il conviendrait de s'interroger sur la nécessité de particulariser deux points de vue. Nous n'avons pas introduit cette notion d'angle dans notre modèle. cela pourrait faire l'objet de futurs développements.

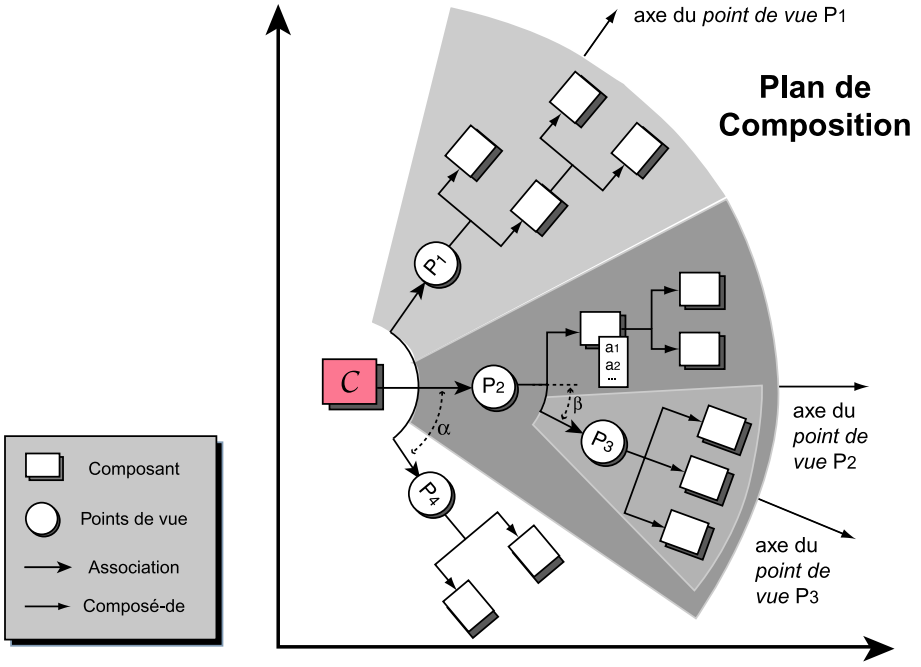


FIG. 2.2 – *Différents points de vue (et sous-points de vue) d'un concept, selon la relation de composition.*

2.3.5 Composant *absent possible*

Les composant *absents possibles* permettent de caractériser des relations de *concomitance* ou d'*exclusion*, particulièrement fréquentes en biologie. Elles traduisent respectivement une condition de présence ou d'absence d'un caractère en fonction du « contexte » formé par d'autres caractères. Noël Conruyt (1994) illustre l'intérêt de prendre en compte ce type de relation :

« . . . Elles interviennent par exemple dans la classification des Mammifères dans le fait qu'ils soient ou non dotés d'un placenta (distinction entre Placentaires et les Aplacentaires) ; il est bien évident que cela ne saurait être observé chez les individus mâles ; si le contexte de la description correspond à un taureau, il est « non pertinent » de s'intéresser à savoir s'il est gravide, ou au nombre de pis portés par ses mamelles » . . . l'information liée à la « non pertinence » du caractère gravide est entièrement portée par le fait que le taureau est de sexe mâle, et traduit le fait général de l'exclusion entre masculinité et gestation. La nature est ainsi faite.

Certains composants sont donc qualifiés d'**absents-possibles**, car le concepteur considère que ces composants peuvent être absents des descriptions. Cela implique en particulier qu'un individu, instance du modèle descriptif, peut appartenir au concept sans pour autant posséder certaines propriétés. Cet ensemble de propriétés, qualifiées de contingentes, est formé des descripteurs en relation de dépendance avec le composant absent-possible. Nous pouvons considérer que ces propriétés dépendantes peuvent être rendues **inapplicables** par l'absence du composant-père ou bien, pour les sous-composants, en état d'**absence possible déduite** [45]. Cet état est implicite, hérité du composant-père, il ne peut pas être explicitement renseigné dans les descriptions car c'est le système qui se charge de propager les états par des règles de contrôle (ou règles de cohérence) [46].

Lorsqu'un composant est absent de la description, tous ses sous-composants doivent également être absents. Un mécanisme de gestion de la cohérence vérifie l'intégrité des dépendances *pères-fils* entre composants, pour garantir d'une part que le *composant-père* d'un composant présent est présent et d'autre part que tous les *composants-fils* d'un composant absent sont absents [47]. Le statut d'un composant peut également être inconnu, lorsque le descripteur [48] ne peut déterminer si le composant est présent ou absent.

Un ensemble de 6 règles de cohérence de la relation *composé-de* et de la relation inverse *partie-de* permet de contrôler l'intégrité de l'état de présence-absence des composants :

Cohérence de la relation père-fils (composé-de)

1. `<absent>` ⟶ `<absent>`
2. `<présent>` ⟶ `<présent> | <absent> | <inconnu>`
3. `<inconnu>` ⟶ `<inconnu> | <absent>`

Justification :

1. les sous-composants d'un composant sont nécessairement absents. C'est l'**absence déduite**. *Un être humain ne peut pas avoir de main gauche s'il n'a pas de bras gauche.*
2. la présence d'un composant ne précise en rien la présence de ses sous-composants. *On n'a pas forcément une main gauche si on possède un bras gauche.*

45. L'absence possible déduite (ou l'absence déduite possible) ne doit pas être confondue avec l'absence déduite. La première concerne les composants du modèle descriptif, alors que la seconde concerne les instances des composants du modèle, donc les descriptions (les cas).

46. on nomme règles de contrôle ou règles de cohérence, selon que l'on considère respectivement la gestion de la dynamique des connaissances ou la cohérence statique de la base.

47. ce que l'on appelle l'**absence déduite**

48. celui qui décrit

3. si l'on ne connaît pas le statut d'un composant, les sous-composants ne peuvent être présents, ce qui impliquerait la présence du composant par la règle 5 ci-dessous.

Cohérence de la relation fils-père (partie-de)

4. <absent> ⟶ <présent> | <absent> | <inconnu>
5. <présent> ⟶ <présent>
6. <inconnu> ⟶ <inconnu> | <présent>

Justification :

4. L'absence d'un sous-composant n'induit aucune contrainte sur la présence de ses sur-composants.
5. la présence d'un sous-composant déduit la présence de ses pères. C'est la **présence déduite**. Si on a une main gauche, on a forcément un bras gauche.
6. si on ne connait pas la présence ou l'absence d'un sous-composant, on peut juste dire que les sur-composants ne sont pas absents, ce qui impliquerait l'absence du sous-composant par la règle 1.

Ces 6 règles offrent donc un mécanisme de contrôle et de maintien de la cohérence des descriptions, par rapport à la sémantique de l'absence possible, dans une dimension horizontale, la composition.

Un sous-arbre intensionnel, plusieurs sous-arbres contingents

Lors d'un parcours récursif en profondeur du modèle descriptif, chaque premier composant *absent possible* rencontré forme la racine d'un sous-modèle (sous-arbre) caractérisant un ensemble de propriétés contingentes. Il existe par contre un seul sous-arbre caractérisant les propriétés nécessaires, de racine identique avec le graphe d'origine.

La plupart des algorithmes exposés par la suite utilisent des parcours d'arbres pour distinguer les propriétés nécessaires et donc systématiquement présentes dans les descriptions, des propriétés contingentes (non-nécessaires), et donc parfois absentes des descriptions.

La figure 2.3 illustre les sous-arbres contingents G_1, G_2, G_3 et G_4 pouvant être extraits de l'arbre complet G correspondant au modèle descriptif total. Les racines respectives de ces quatre sous-arbres sont les composants (ou les points de vue) absents possibles : C_9, C_2, C_7 et P_2, marqués d'un signe **-** (gras). Notons que C_3, C_4, C_5 et C_6, marqués d'un signe - (normal), ne possèdent que le statut d'absence possible déduite et ne peuvent être racine d'un sous-graphe contingent : il ne peuvent être absents que si leur ancêtre absent possible est absent (absence déduite). Le statut associé aux attributs est donné par celui du composant auxquels ils sont attachés : a_1 et a_2 de C_8 sont nécessaires, a_1 et a_2 de C_3 sont contingents.

La donnée du sous-arbre intensionnel offre une base commune pour comparer les descriptions, dont certaines parties peuvent être absentes. De plus, lorsque plusieurs experts travaillent sur le même groupe taxonomique, ils peuvent développer parallèlement deux modèles descriptifs de même niveau, en s'accordant au préalable sur la structure du sous-arbre nécessaire. Ils ont ainsi la liberté de définir différents points de vue et sous-modèles contingents, selon leur conception

personnelle du domaine. Les descriptions sont alors compatibles, tout au moins sur la partie commune, c'est-à-dire le sous-arbre intensionnel.

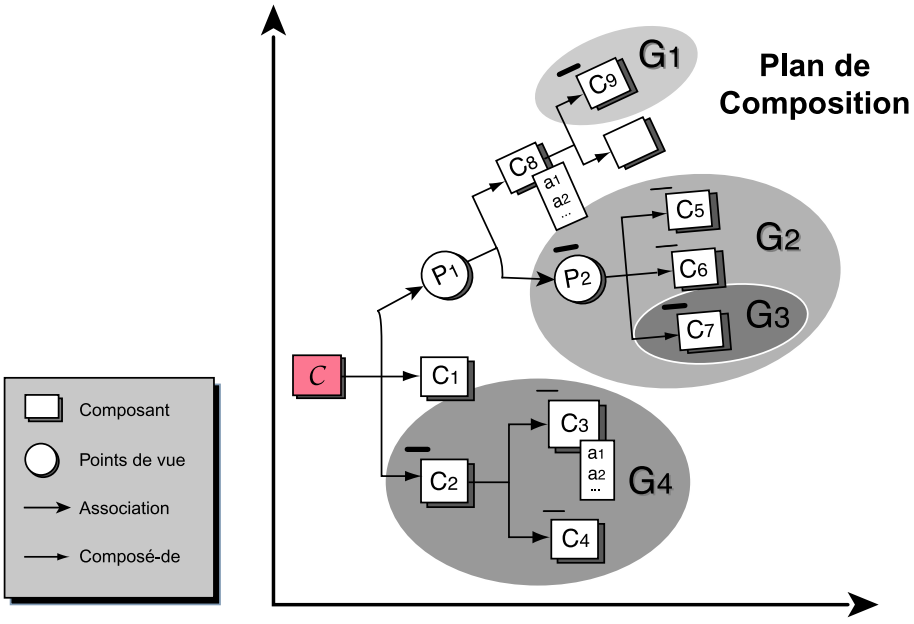


FIG. 2.3 – *Un sous-arbre intensionnel, plusieurs sous-arbres contingents*

2.3.6 Multiplicité

Un composant peut également être **multiple** et posséder une propriété de multiplicité exprimée sous la forme d'un intervalle d'entiers $[inf\,sup]$, avec $inf \geq 1$. Un composant multiple peut être instancié plusieurs fois dans la description des cas (au maximum, la borne supérieure de la multiplicité et au minimum la borne inférieure). Par exemple, un composant de multiplicité [1 4] pourra être instancié jusqu'à quatre fois dans les descriptions. Par défaut, lorsqu'aucune multiplicité n'est exprimée, un composant est instanciable une fois.

La multiplicité correspond à la *logique d'itération* mise en avant par Conruyt (1994) dans « les logiques descriptives en sciences de la vie » comme un moyen permettant d'exprimer la variabilité dans les observations pour un objet d'étude particulier (symbolisé par un composant), c'est-à-dire le moyen de renseigner plusieurs états observés simultanément pour un même objet.

L'exemple suivant illustre l'utilité de cette logique d'itération. Supposons que l'objet de notre étude s'attache à la description locale de l'inflorescence [49] (voir figure 2.4). Plus précisément, le cas où l'inflorescence est définie, c'est-à-dire qu'elle se termine généralement par une fleur (Roche et al. 2000).

Admettons qu'un unique caractère soit modélisé, la couleur des fleurs avec comme liste de couleurs possibles *blanc*, *jaune* et *rouge*. Si deux couleurs sont observées (pour des fleurs différentes), il est souhaitable de pouvoir exprimer cette variabilité par un fait conjonctif $blanc \wedge jaune$ traduisant la présence simultanée de la couleur *blanc* et *jaune* sur les fleurs, plutôt que par un

49. **Inflorescence** : appareil reproducteur d'une plante

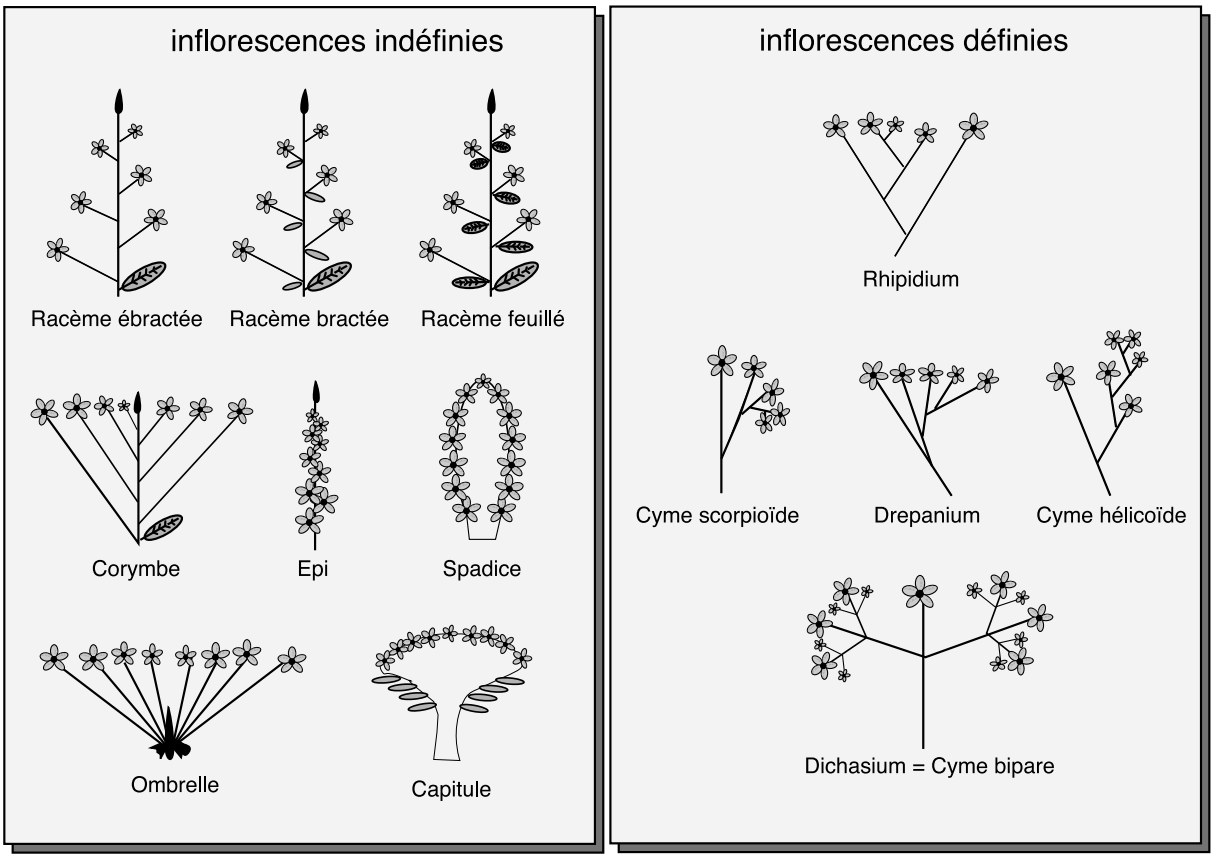


FIG. 2.4 – *Deux sortes d'inflorescences : définies ou indéfinies (Roche et al. 2000).*

fait disjonctif, exprimé par l'ensemble $\{blanc,jaune\}$, équivalent à $blanc \vee jaune$, qui est interprété comme une imprécision, c'est-à-dire l'incapacité du descripteur à déterminer précisément la couleur de l'ensemble des fleurs. Considérons maintenant que l'inflorescence est réifiée [50] par un composant, nommé *Inflorescence*, qui possède plusieurs attributs, tels que la *forme*, la *taille*, la *couleur* et le *type*. Affecter une multiplicité à ce composant (par exemple [1 4]) permet d'instancier plusieurs fois ce composant, jusqu'à 4 fois, de renseigner séparément chacun des attributs et donc de décrire différentes sorte d'inflorescences sur une même plante sans être forcé de les nommer. Notons enfin que c'est l'expert qui prédéfinit le nombre maximal d'instances dans le modèle descriptif, en fonction du nombre de sorte possibles d'objets et la connaissance de leurs noms.

La figure 2.5 illustre un composant de multiplicité [1 4] instancié deux fois dans la description d'un cas. Deux formes d'inflorescence sont décrites, *scorpioïde* et *hélicoïde* (Roche et al. 2000), sans qu'il soiit nécessaire de donner le type plus précis de l'inflorescence (défini ou indéfini).

50. **Réifier** v.t. : Fait de transformer en chose ce qui est mouvant, dynamique, ou ce qui est de l'ordre de la simple représentation mentale. *Le Petit Larousse, éd. 1998*

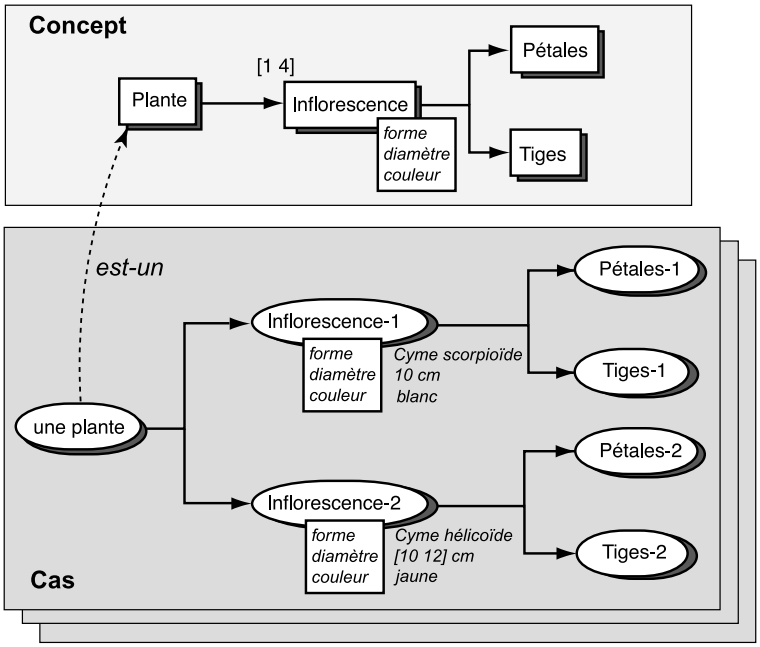


FIG. 2.5 – *L'objet Inflorescence, de cardinalité* [1 4] *au niveau de l'observable est instancié (décrit) deux fois au niveau d'une observation.*

2.3.7 Polymorphisme de composants

Les composant peuvent également être *spécialisés* pour distinguer les différentes sortes d'objets, du plus général au plus précis. A partir de la définition d'un composant général, une hiérarchie de spécialisation (simple) peut être formée en nommant des composants plus spécialisés.

Le sens accordé à la spécialisation de composants est quelque peu différent de celui accordé à la relation de spécialisation dans les modèles objets. Deux règles générales définissent la spécialisation de composants dans notre modèle :

Définition 2.2 *Règles de spécialisation des composants :*

1. *un composant possède toutes les caractéristiques* ***nécessaires*** *du composant dont il hérite.*
2. *un composant possède, par union généralisée, toutes les caractéristiques des composants qui le spécialisent.*

La justification précise de la spécialisation dans le modèle CoDesc est donnée au § 2.6 qui traite des hiérarchies de concepts. L'interprétation de la spécialisation des composants et des concepts est la même : un composant définissant en quelque sorte un concept localisé (un sous-concept) à une partie du concept plus général, le modèle descriptif. Nous nous intéressons dans cette partie, au rôle de la spécialisation des composants au sein d'un modèle descriptif du point de vue de la représentation des connaissances en biologie.

Par exemple, la nature de l'inflorescence peut-être spécialisée en *définie* ou *indéfinie*. Ainsi, le fait de définir deux composants correspondant aux deux sortes d'inflorescences, permet de distinguer deux sous-modèles descriptifs pour lesquels d'une part les fleurs seront décrites (inflorescence définie) et d'autre part non décrites (pas de fleurs pour les inflorescences non définies).

La figure 2.6 illustre une modélisation possible de l'inflorescence. Le composant *Inflorescence* défini au niveau du modèle est un objet *polymorphe*, qui est spécialisé en deux composants, *Inflorescence définie* et *Inflorescence indéfinie*. Il peut être instancié jusqu'à 4 fois (cardinalité [1 4]). Selon la règle 1 de la définition 2.2, les deux composants spécialisés héritent des propriétés nécessaires de *Inflorescence*, le composant *Plante*, l'attribut *forme* dont le domaine de définition est restreint aux formes possibles selon le type d'inflorescence : *Racème*, *Corymbe*, *Epi*, *Spadice* ou *Ombrelle*, pour les inflorescences indéfinies et *Cyme*, *Drepanium*, pour les inflorescences définies (Roche et al. 2000). *Fleur* étant un composant *absent possible*, il n'est pas nécessairement hérité : il est présent dans *Inflorescence définie* mais pas dans *Inflorescence indéfinie*. Dans la description du cas, le composant *Inflorescence* est instancié trois fois. Dans la première instance, l'observateur n'a su déterminer précisément le type de l'inflorescence ni la forme, mais a néanmoins décrit une fleur. Cette information suffit au système pour effectuer la spécialisation de manière automatique.

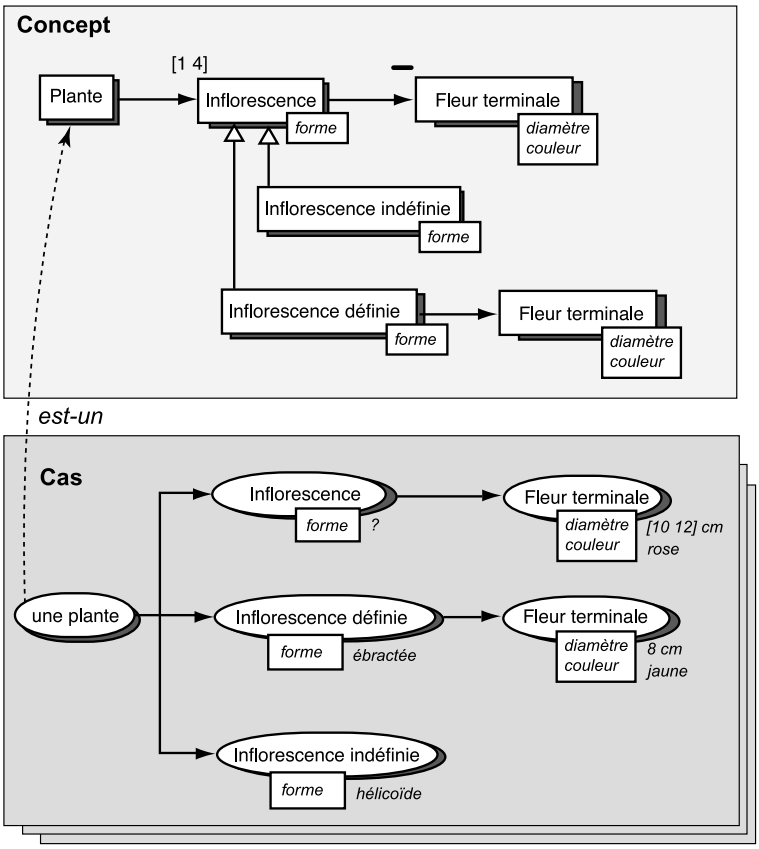


FIG. 2.6 – *Association de la multiplicité et de la spécialisation d'un composant.*

La spécialisation de composants présente un grand intérêt lorsqu'il est difficile de définir *a priori* la nature d'un objet observé, du fait du *polymorphisme* de celui-ci. Dans les sciences de la nature en particulier, il est fréquent de ne pouvoir nommer explicitement ce que l'on voit à un niveau de précision suffisant. Il arrive souvent que, dans une description, plusieurs caractères, bien que non rigoureusement identiques, soient de même type mais présentent des caractéristiques différentes.

Noël Conruyt (1994) propose l'illustration suivante pour montrer l'intérêt de la spécialisation

de composants :

« *. . . Prenons par exemple les dents des mammifères. Si nous avons à décrire la denture d'un homme, nous observons qu'il existe plusieurs sortes de dents, disons 3 ou 4 sortes selon notre perspicacité. Les plus savants les désignerons d'emblée : incisives, canines, prémolaires et molaires* . . . »

Une modélisation possible consiste dans cet exemple à définir un objet général *dent*, de multiplicité [1 4] permettant de décrire jusqu'à quatre instances de dents, ainsi que quatre composants spécialisés correspondant aux quatre sortes de dents existantes. Ainsi, un observateur pourra instancier jusqu'à quatre types de dents puis les décrire et/ou les spécialiser en fonction de sa connaissance du domaine.

La création de hiérarchies locales d'objets permets de respecter le principe d'*homologie*. Si nous devons comparer dans le détail les dentures du chien et du chat, il faut s'assurer que nous comparons bien les canines (ou "crocs") de l'un avec les canines de l'autre. Sinon, le principe n'est pas respecté. « *. . . Il faut être conscient du risque d'interprétation (donc de subjectivité) qu'il peut y avoir à s'aventurer dans des "déterminations locales d'objets" ; le descripteur, non averti des limites de son savoir désignerait comme des canines les défenses du Morse et celles de l'éléphant commettrait une erreur, qui par suite conduirait à comparer des objets non véritablement homologues : les défenses de l'Eléphant sont des incisives modifiées, contrairement à celles du Morse qui sont bien des canines, quoique d'une taille exceptionnelle.* »

La définition d'un objet polymorphe au niveau du modèle descriptif selon la sémantique particulière accordée à la spécialisation des composants dans CoDesc, offre plusieurs avantages pour décrire une observation :

- Renseigner l'ensemble des descripteurs sans connaître précisément le type d'un objet. Au niveau de B, toutes les caractéristiques sont accessibles.
- La présence d'un objet particulier permettra de déduire le type précis de l'objet décrit. Par exemple, la présence de fleur terminale dans la description de l'inflorescence permet de déduire que le type de cet objet est une *Inflorescence définie*.

2.4 Les attributs

Les attributs sont attachés aux composants ou au schéma du modèle descriptif et définissent des parties du modèle. Les attributs sont simples et ne peuvent référencer d'autres composants ni d'autres attributs.

Nous notons $\mathcal{A}$, l'ensemble des attributs d'un modèle descriptif et $A \in \mathcal{A}$ un attribut particulier. A est défini par un ensemble de propriétés qui précisent son *type*, possède un *identificateur* qui permet de le référencer de manière unique dans la base et un *domaine de définition*, noté $\mathbf{dom}(A)$ spécifiant l'ensemble des valeurs simples admissibles par les instances. Un attribut $A \in \mathcal{A}$ peut être vu comme une fonction qui associe à tout objet une ou plusieurs valeurs. Cette fonction a pour ensemble de définition Ω, et pour co-domaine, l'ensemble des valeurs admissibles par les instances, noté $\mathbf{cod}(A)$. En particulier, la valeur inconnue est admissible pour toute instance. Les ensembles de valeurs de $\mathbf{dom}(A)$ sont également autorisés pour toute instance, sans qu'aucune restriction ne soit précisée.

Nous reviendrons sur cet aspect, mais pour fixer les idées, on considère que, pour les descriptions, un attribut A est une application à valeur dans l'ensemble des parties du domaine, noté $\mathcal{P}(\mathbf{dom}(A))$. Cette définition permet de considérer que toute valeur valide de A peut-être un sous-ensemble de valeurs de $\mathbf{dom}(A)$, un singleton, l'ensemble vide ou le domaine tout entier. La valeur inconnue est considérée comme une valeur particulière qui généralise toute valeur possible que peut prendre une instance. Dans cette acception, les attributs ne précisent pas de contraintes sur la cardinalité de l'ensemble de valeurs simples du domaine que peuvent prendre les observations : tout attribut, quel que soit son type, peut être *mono-valué* ou *multi-valué*.

Il est possible de spécifier une *valeur par défaut* qui sera prise par les instances à défaut de la valeur vide (ou ensemble vide). En général, la valeur la plus fréquente est choisie.

Notons que le statut de propriété contingente ou nécessaire n'est pas explicite pour les attributs : il est déduit du composant auquel il est attaché. Si le composant est *absent possible* (ou absent possible déduit), toutes ses propriétés sont contingentes (y compris les règles). Le modèle ne permet donc pas de distinguer différents statuts possibles pour les attributs d'un même composant. Lorsque l'on souhaite effectuer cette distinction, il faut procéder à une décomposition supplémentaire du composant en deux sous-composants attachant respectivement le sous-groupe de propriétés contingentes et le sous-groupe de propriétés nécessaires.

2.4.1 Les types d'attributs

La complexité des observations issues du monde réel implique la nécessité de disposer de différents types d'attributs pour décrire ces observations. On aura besoin par exemple qu'un attribut prenne des valeurs *numériques*, pour décrire une longueur ou un poids, symboliques pour exprimer l'aspect ou la forme, ou bien hiérarchisées pour qualifier différents niveaux de précisions de l'observation. Nous présentons dans cette partie les différents types d'attributs utilisés dans le modèle CoDesc. L'étude des valeurs que peuvent prendre les attributs dans les observations fera l'objet du § 2.9.

Le *type* d'un attribut décrit la nature du domaine de définition de l'attribut, en particulier la structure sous-jacente de l'ensemble : ordre total ou partiel entre valeurs, les bornes du domaine, etc. Il conditionne également les opérations possibles sur les valeurs et sur les ensembles de valeurs (union, intersection, généralisation, borne inféreur, etc.). Les types sont prédéfinis et ne peuvent être créés, étendues ou modifiés par l'utilisateur de la base. La plate-forme $\mathcal{IKBS}$ offre cependant des mécanismes de création de nouveaux types, sous la forme d'interfaces de conception ou par spécialisation de types existants, qui facilitent l'extension du modèle par le programmeur (cf. chap. 6).

Plus formellement, la définition d'un attribut est donnée par un type et un domaine :

Définition 2.3 *Soit $\mathcal{A} = \{A_1, \ldots, A_p\}$, l'ensemble fini de tous les attributs définis dans le modèle descriptif, dénotant un concept $\mathcal{C}$.*

Soit $\mathcal{D} = \{D_1, \ldots, D_p\}$, l'ensemble fini des domaines (notés également $\mathbf{dom}(A)$) et $\mathcal{T} = \{\tau(A_1), \ldots, \tau(A_p)\}$, l'ensemble des types associés au domaine de chaque attribut de $\mathcal{A}$ à valeur dans {numérique, nominal, ordinal, booléen, taxonomique, textuel}.

Un attribut A_i est défini formellement par le triplet (A_i, D_i, $\tau(A_i)$).

Nous regroupons sous le terme générique type *symbolique*, tout attribut dont le domaine est caractérisé par un ensemble de symboles. Les symboles représentent les valeurs simples qu'une observation est susceptible de prendre. Le type symbolique regroupe les types nominal, ordinal, booléen et taxonomique.

Type nominal

Pour un type symbolique, lorsqu'aucune relation d'ordre n'existe entre les symboles, l'attribut est de type *nominal*.

Un attribut symbolique possède un domaine caractérisé par un ensemble de symboles. Ces symboles représentent les valeurs simples qu'une observation est susceptible de prendre. Un domaine symbolique peut être simplement un ensemble de symboles, sans relation d'ordre, le type est alors *nominal*. Il peut être muni d'une relation d'ordre total $\leqslant$, il est alors de type *ordinal*.

Exemple de descripteur nominal : $Pays = \{France, Chine, Suisse, Yougoslavie, \ldots\}$. Aucune relation d'ordre n'existe entre les valeurs.

Un cas particulier de type nominal est le type booléen, lorsque le domaine possède deux valeurs $\{oui, non\}$, $\{vrai, faux\}$, {0, 1}, etc.

Type ordinal

Lorsqu'un type symbolique est muni d'une relation d'ordre entre les symboles, notée $\leqslant$, le type est ordinal.

Exemple de descripteur ordinal : $Longueur = \{Petit, Moyen, Grand\}$. La présence d'une relation d'ordre entre les valeurs permet de considérer que la valeur $Petit$ est plus proche de $Moyen$ que de $Grand$, ce qui est utile pour comparer les observations : $Petit \leqslant Moyen \leqslant Grand$.

Type taxonomique

Le domaine d'un attribut symbolique peut être structuré sous la forme d'une taxonomie de valeurs. La taxonomie[51] permet d'ordonner partiellement les différentes valeurs d'un domaine de type symbolique.

La figure 2.7 illustre un ensemble de couleurs selon différents niveaux de précision, à l'aide d'une taxonomie de valeurs. Par exemple, la valeur *clair* est plus générale que *rose*, *blanc* et *jaune*. En particulier, la valeur *indéterminé* n'offre aucune précision, c'est la valeur la plus générale, racine de la taxonomie de valeurs. Elle est équivalente à la valeur *inconnu*.

Type numérique

Le domaine de valeur d'un attribut numérique est défini par un intervalle de $\mathbb{R}$, noté noté [inf, sup]. Toute valeur valide pour une observation doit être incluse dans cet intervalle. Comme pour

51. Taxonomie ou taxinomie.

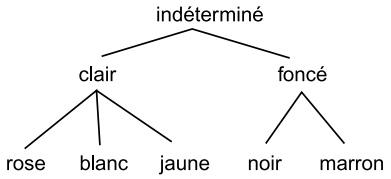


FIG. 2.7 – *Exemple d'une taxonomie pour la couleur*

tous les types considérés, un domaine numérique est potentiellement multi-valué, c'est-à-dire que tout intervalle $I \subseteq [inf,sup]$ est une valeur valide.

Type textuel

Le type textuel permet d'entrer toute *chaîne de caractères* comme valeur possible des descriptions. Il ne définit aucune contrainte sur cette entrée. Les attributs textuels sont utiles pour renseigner certaines parties des descriptions, comme le nom de l'auteur, la date de création, etc. Notons que les attributs textuels n'interviennent jamais dans les méthodes d'analyse.

Les attributs textuels sont également utiles lors de la conception d'un modèle descriptif. En effet, plutôt que de définir dès le début un descripteur symbolique, qui contraint les observations à prendre des valeurs dans un espace prédéfini, le concepteur peut choisir un type textuel, ce qui laisse l'observateur libre de nommer les valeurs comme il l'entend. Dans un deuxième temps, le concepteur « transforme » l'attribut en type symbolique et le système se charge de construire automatiquement le domaine de définition symbolique, à partir des termes présents dans la base de cas. Nous reviendrons sur les opérations de changements de types lors de l'étude du mécanisme de *mutation d'objets* aux chapitres 6.

2.5 Les règles

Des règles peuvent être attachées aux composants et aux attributs du modèle. Une règle autorise deux types d'actions :

- des **contraintes**, pour éviter par exemple qu'une valeur particulière soit affectée à un descripteur, en fonction de l'état d'autres descripteurs, ou bien pour forcer la valeur d'un descripteur,
- des liens **fonctionnels** entre descripteurs, pour permettre une inférence de valeurs ou un calcul résultant d'opérations logiques ou arithmétiques à partir des valeurs d'autres descripteurs.

D'une façon générale, les règles de contrainte permettent de réduire l'extension associées au concept, pour certains cas particuliers relevé par l'expert. Elles peuvent en particulier permettrent d'interdir certaines valeurs lorsque qu'une instance appartient à une classe donnée. Par exemple, l'expert souhaite signifier une caractéristique marquante de l'espèce *Pocillopora verrucosa*, qui ne possède jamais de septes directeurs. La règle suivante pourra être attachée à l'attribut *taxon*

de l'objet identification :

```
SI [taxon[identification] = Pocillopora verrucosa] ALORS
[septes-directeurs = ABSENT]
```

Les règles fonctionnelles sont utiles pour effectuer des opérations arithmétiques, comme par exemple la valeur du `nombre[septes]` qui est calculée automatiquement par le système de la façon suivante :

```
nb[septes] = nb[septes-primaires] + nb[septes-secondaires] + nb[septes-tertiaires]
```

La valeur du `nombre[septes]` est alors calculée automatiquement par le système.

Activation des règles

Les règles de contrainte sont attachées à chaque élément (composant ou attribut) référencé dans la prémisse, soit l'attribut `taxon[identification]` dans le premier exemple. Ainsi, lors de la description d'une observation, lorsqu'une modification est effectuée sur un des attributs référencé dans la partie prémisse de la règle, celle-ci est activée et la valeur inférée est affectée à l'attribut situé dans la partie conclusion de la règle. Le principe est similaire pour les règles fonctionnelles.

2.6 Spécialisation et généralisation de concepts

2.6.1 La classification naturelle

L'homme organise naturellement les concepts (correspondant à des catégories d'individus) sous la forme de *hiérarchies*, où des concepts plus généraux dominent des concepts plus spécifiques (Napoli 1992). Cette classification naturelle est établie par des spécialistes (les systématiciens) à partir de l'étude d'un ensemble de spécimens appartenant à des niveaux taxonomiques différents. Un grand nombre de critères morphologiques sont considérés traditionnellement, par observation directe (oeil, loupe, microscope) et d'autres méthodes et examens sont utilisés plus récemment pour obtenir d'autres critères non visuels : techniques embryologiques, biochimiques, histologiques, cytologiques, génétiques ou radiologiques, etc. [52]

Dans les classifications naturelles, botaniques, zoologiques ou minérales, se distinguent généralement six grands niveaux :

- Le **Règne** est animal, végétal, ou minéral. Il représente la division la plus large du monde naturel ;
- L'**Embranchement** est la grande division du règne, essentiellement pour les règnes végétal et animal. Un embranchement peut éventuellement avoir des sous-embranchements ;

52. Voir par exemple le projet MedLine(de Rouen 2001) pour un thesaurus hiérarchisé des différentes techniques de recherches utilisés en biologie. Notons que le modèle descriptif peut prendre en compte ces autres critères non phénétiques au sein de sous-arbre descriptifs indexés par ces points de vue.

- La **Classe** regroupe des individus, êtres, objets, faits, en nombre indéterminé, et possédant un ou plusieurs caractères communs. Ainsi, la classe des oiseaux et celle des mammifères font partie du sous-embranchement des vertébrés. L'**Ordre** est la première subdivision de la classe ;
- La **Famille** regroupe des genres ayant des traits généraux communs ;
- Le **Genre** regroupe des individus qui sont morphologiquement semblables ;
- L'**Espèce** regroupe des catégories d'êtres vivants, végétaux ou animaux, qui sont caractérisés par des formes bien définies et qui constituent un type héréditaire de fécondité illimitée par croisement (espèces naturelles).

A ces six catégories, certains auteurs définissent parfois une sous-catégorie de l'espèce, appelée **sous-espèce** ou **écomorphe**.

En ce qui concerne les coraux par exemple, certaines espèces présentent des écomorphes, lorsque la forme de l'espèce est très dépendante des conditions du milieu (mode, éclairement, turbidité, température)(Veron et al. 1976) et présente une forte variabilité intra-spécifique. Par exemple, l'espèce *Pocillopora damicornis*, « corail dur » présent dans les Mascareignes appartient aux catégories suivantes (d'après Faure 1982) :

```
Classe      : Anthozoa        (Ehrenberg, 1834)
Sous-classe : Zoantharia      (De Blainville, 1830)
Ordre       : Scleractinia    (Bourne, 1905)
Sous-ordre  : Astrocoeiina    (Vaughan & Wells, 1943)
Famille     : Pocilloporidae  (Gray, 1842)
Genre       : Pocillopora     (Lamarck, 1816)
espèce      : damicornis      (Linné, 1758)
```

Elle présente une variabilité intra-spécifique importante et de ce fait peut-être spécialisée en cinq écomorphes en fonction des conditions du milieu :

```
Ecomorphe bulbosa (formes de mode calme à très calme)
Ecomorphe acuta (récif interne, vasque de mode très calme)
Ecomorphe favosa (mode calme, zone d'hper-sédimentation)
Ecomorphe brevicornis (mode battu à très battu)
Ecomorphe caespitosa (biotopes à éclairement réduit)
```

2.6.2 Organisation verticale des concepts

L'organisation des concepts en une hiérarchie conceptuelle relève d'une *dimension verticale* de la structuration des connaissances, c'est-à-dire qu'elle traite de l'ordonnancement des catégories les unes par rapport aux autres (Napoli 1992), ce qui relève de l'organisation externe ou encore de la classification des catégories (Rosch et al. 1976).

Dans le modèle CoDesc, les concepts[53] sont regroupés au sein d'une base de concepts qui constitue le noyau de la base de connaissances. Ils sont organisés par l'intermédiaire d'une relation

53. par l'intermédiaire de leur représentation informatique : les modèle descriptifs.

de spécialisation (ou relation d'héritage) $\preccurlyeq$ en une hiérarchie conceptuelle $\mathcal{H}$. L'élément maximal de la hiérarchie est noté $\mathcal{C}_0$. Chaque concept ne possède qu'un seul ascendant, l'héritage est donc simple et $\mathcal{H}$ forme une structure arborescente, appelée *arbre de spécialisation*.

La relation de spécialisation définit un ordre partiel sur l'ensemble des concepts. Un concept hérite obligatoirement de toutes les **propriétés nécessaires** de ses ascendants : les concepts d'ordre supérieur ou *super-concepts* (fig. 2.8). Les propriétés contingentes peuvent ou non être héritées. Nous détaillerons précisément ci-dessous le sens particulier que nous accordons à la spécialisation, du fait de la présence de propriétés nécessaires/contingentes.

Les propriétés sont héritées statiquement : elles sont recopiées dans les sous-concepts et spécialisées, c'est-à-dire que les domaines de valeurs doivent être restreints. Nous verrons plus bas que la spécialisation, et la relation inverse, la généralisation, sont **transitives**, **réflexives** et **antisymétriques**.

Dans l'exemple illustré par la figure 2.8, les concepts représentés sont réduits à la seule expression de deux attributs, notés a (numérique) et b (symbolique). Tout concept hérite de l'attribut a, nécessaire, alors que b, absent possible au niveau de $\mathcal{C}_1$, est absent de $\mathcal{C}_2$. Il est par contre nécessaire dans $\mathcal{C}_3$. Les instances (individus w_1,w_2 et w_3) sont rattachées à un unique concept qui définit l'ensemble de leurs propriétés et sont donc incluses dans l'extension des concepts. Notons qu'une instance de $\mathcal{C}_0$ ou de $\mathcal{C}_1$ peut ne pas posséder la propriété b.

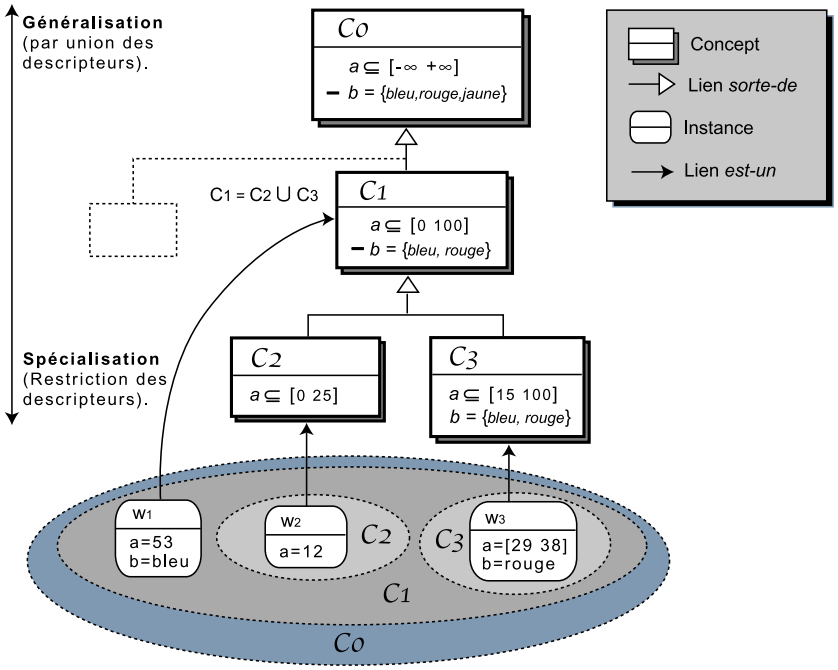


FIG. 2.8 – *Hiérarchie de concepts dans CoDesc.*

Inversement, en considérant la relation opposée de la spécialisation, tout concept *généralise*, par union, la totalité des descripteurs de ses sous-concepts. Ainsi, dans la figure 2.8, le concept $\mathcal{C}_1$ généralise les propriétés de ses sous-concepts $\mathcal{C}_2$ et $\mathcal{C}_3$. b est une propriété *absente possible* dans $\mathcal{C}_1$, car elle est absente de la définition de $\mathcal{C}_2$.

Plus précisément, comme nous l'avons évoqué lors de la partie traitant du polymorphisme de

composant, l'origine des propriétés qui définissent un concept provient :

- De l'ensemble des propriétés nécessaires, héritées de son ascendant. Ce qui correspond à la logique de la spécialisation des systèmes à objets. Toute propriété est cependant héritée statiquement pour être éventuellement spécialisée, par restriction des domaines.
- De l'union généralisée de toutes les propriétés de ses sous-concepts. Les propriétés qui ne sont pas communes à tous les sous-concepts [54], sont contingentes.

Notons que la conception d'une taxonomie de concepts dans CoDesc s'effectue de préférence par une approche ascendante. Un concept est construit par généralisation des concepts qu'il subsume (en utilisant l'opérateur d'union généralisé), plutôt que par spécialisations successives des concepts. Il est en effet très difficile de définir *a priori*, à un niveau de granularité élevé, l'ensemble des caractères pouvant intervenir dans les descriptions.

2.6.3 Nécessité, contingence et subsomption

Un concept hérite de toutes les propriétés de son ascendant, qu'il peut éventuellement spécialiser : une propriété contingente (marquées d'un signe - dans l'exemple de la figure 2.8) peut ne pas exister ou au contraire devenir nécessaire. Par exemple, le concept $\mathcal{C}_2$ n'hérite pas de l'attribut b défini dans $\mathcal{C}_1$, alors que b est nécessaire dans $\mathcal{C}_3$. Les domaines de valeurs peuvent être restreints : $\mathbf{dom}(b) = \{bleu, rouge, jaune\}$ dans $\mathcal{C}_0$ et $\mathbf{dom}(b) = \{bleu, rouge\}$ dans $\mathcal{C}_3$. Pour éviter toute confusion, nous pouvons préfixer les attributs par le nom du concept auquel il est attaché, de la façon suivante : $\mathcal{C}_0.b$ et $\mathcal{C}_2.b$. Les propriétés nécessaires se retrouvent obligatoirement dans les sous-concepts et ne peuvent devenir contingentes. De même, les domaines de valeur des attributs ne peuvent étendre le domaine de l'attribut hérité.

Inversement, en poursuivant un ordre ascendant, un concept généralise toutes les propriétés de ses sous-concepts. Une propriété nécessaire à un niveau donné peut-être nécessaire ou contingente au niveau supérieur.

Pour résumer, nous donnons un aperçu des règles de transformation du statut d'une propriété, selon le sens de la spécialisation et celui de la relation inverse, la relation de généralisation :

Spécialisation

Les règles suivantes relient le statut d'une propriété d'un concept au statut de la même propriété spécialisée dans les sous-concepts.

```
1. <nécessaire>   ⟶  <nécessaire>
2. <contingente>  ⟶  <contingente> | <nécessaire> | <absente>
3. <absente>      ⟶  <absente>
```

La figure 2.9 illustre les règles de transformation des trois états possibles des descripteurs, par spécialisation.

54. c'est-à-dire qui n'appartiennent pas à l'intersection de tous les concepts

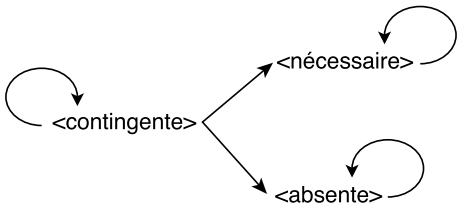


FIG. 2.9 – *Automate fini des 3 états possibles des descripteurs et de leur transformations par spécialisation.*

Généralisation

Les règles suivantes relient le statut d'une propriété définie dans un concept à la même propriété généralisée dans son super-concept.

```
4. <nécessaire>   ⟶  <contingente> | <nécessaire>
5. <contingente>  ⟶  <contingente>
6. <absente>      ⟶  <absente> | <contingente>
```

La figure 2.9 peut être interprétée pour la généralisation, en inversant le sens des relation de changement d'état. En effet, les règles 4, 5 et 6 sont simplement obtenues en inversant les règles 1, 2 et 3. En inversant 1, 2 et 3, on obtient :

```
1'. <nécessaire>   ⟶  <nécessaire>
2'. <contingente>  ⟶  <contingente>
3'. <nécessaire>   ⟶  <contingente>
4'. <absente>      ⟶  <contingente>
5'. <absente>      ⟶  <absente>
```

Ainsi, [1' et 3' produisent 4], [2' produit 5] et [4' et 5' produisent 6].

Justification et interprétation des règles :

1. Un concept possède obligatoirement les propriétés nécessaires du concept qui le subsume. En effet, si l'on dit que *tous les oiseaux ont des plumes* est une condition nécessaire du concept `Oiseau`, on ne peut définir de sous-concept (sous-catégorie) d'oiseaux qui n'ont pas de plumes, cela reviendrait à dire que ce « drôle d'oiseau » n'est pas vraiment un oiseau (comme le grand pingouin). Dans ce cas, il est préférable de préciser que cette propriété est contingente, et de l'exprimer par *la plupart des oiseaux ont des plumes.*
2. La propriété contingente *la plupart des oiseaux ont des plumes* peut se spécialiser en « cette sorte d'oiseaux a des plumes » (regroupe la plupart des oiseaux) et « cette sorte d'oiseau n'a pas de plume » (les pingouins) et donc absence de la propriété. Elle peut également rester contingente.
3. Une nouvelle propriété ne peut pas être introduite dans un sous-concept. L'espace de description des concepts est délimité par celui du super-concept.
4. Une propriété nécessaire dans un sous-concept peut devenir contingente dans un super-concept. En effet, si deux concepts $\mathcal{C}_1$ et $\mathcal{C}_2$ possèdent la même propriété *cette sorte d'oiseau*

vole, nécessaire dans $\mathcal{C}_1$ et contingente dans $\mathcal{C}_2$, qui s'interprète : *la plupart des oiseaux de* $\mathcal{C}_2$ *volent* (certains d'entre eux ne volent pas). Cette propriété est également contingente dans le concept $\mathcal{C}_0$ qui les subsume, c'est-à-dire que *la plupart des oiseaux de* $\mathcal{C}_0$ *volent* puisqu'en particulier certains oiseaux de $\mathcal{C}_2$ ne volent pas. Par contre, si la propriété est nécessaire dans tout sous-concept, elle l'est aussi dans le super-concept.

5. L'interprétation de cette règle est la même que 4.
6. Une propriété absente ne peut pas être nécessaire dans le super-concept à cause de la règle 1. Elle est donc soit absente (mais existe-t-elle ?), soit contingente.

Cette façon de concevoir une hiérarchie peut paraître déroutante, (en particulier la règle 3) dans la mesure où la plupart des modèles hiérarchiques (en particulier les modèles objets) procèdent par ajout de propriétés dans les sous-classes, ce qui n'entraîne aucune modification des sur-classes. Pratiquement, dans le modèle CoDesc, lors de la construction d'un modèle descriptif, des propriétés sont effectivement ajoutées, mais elles doivent alors être généralisées dans le concept ascendant par les règles 4, 5 et 6. Inversement, elles sont héritées par les sous-concepts selon le sens donné par les règles 1, 2 et 3.

La spécialisation et la généralisation définissent un ordre partiel ($\preccurlyeq$) sur les concepts. Elles sont réflexives, transitives et antisymétriques.

- **réflexive**, un concept est sous-concept de lui-même, ce qui est en accord avec les règles de transformation : les propriétés peuvent garder le même statut lors de la spécialisation et donc lors de la généralisation. $\mathcal{C} \preccurlyeq \mathcal{C}$
- **transitive**, qui s'interprète : si $\mathcal{C}_2$ est un sous-concept de $\mathcal{C}_1$, $\mathcal{C}_1$ un sous-concept de $\mathcal{C}_0$, alors $\mathcal{C}_2$ est un sous-concept de $\mathcal{C}_0$. $\mathcal{C}_1 \preccurlyeq \mathcal{C}_2$ et $\mathcal{C}_2 \preccurlyeq \mathcal{C}_3 \Rightarrow \mathcal{C}_1 \preccurlyeq \mathcal{C}_3$
- **antisymétrique**, si $\mathcal{C}_1$ est un sous-concept de $\mathcal{C}_0$, alors $\mathcal{C}_0$ n'est pas un sous-concept de $\mathcal{C}_1$. En particulier, si une propriété contingente devient nécessaire (règle 2), elle ne peut devenir à nouveau contingente (règle 1). $\mathcal{C}_0 \preccurlyeq \mathcal{C}_1$ et $\mathcal{C}_1 \preccurlyeq \mathcal{C}_0 \Rightarrow \mathcal{C}_0 = \mathcal{C}_1$

La figure 2.10 illustre une hiérarchie de concepts selon un plan horizontal et un plan vertical correspondant (resp.) à la relation de composition et de généralisation (ou subsomption). Le composant A_2 de $\mathcal{C}_2$, hérité du composant A_1 de $\mathcal{C}_1$ devient nécessaire, ce qui est conforme avec les règles 2 et 4. Le composant E_2 devient absent (règles 2 et 6). Le composant B_2 reste nécessaire (règles 1 et 4) et D_2 reste contingent (règles 2 et 5).

2.7 Les classes

Les classes regroupent des sous-ensembles significatifs d'instances d'un concept. Dans le modèle CoDesc, les classes ne sont pas réifiées en tant qu'entités du modèle, elles sont définies par le domaine d'un attribut particulier d'un concept : **l'attribut de classe**[55]. En systématique, cet attribut est généralement appelé *taxon* car les classes considérées sont principalement des noms de taxons. Le choix de l'attribut de classe est laissé au concepteur de la base, mais doit être nécessairement de type discret. Cet attribut peut être éventuellement muni d'une relation

55. l'attribut de classe peut également être appelé caractère *diagnostic*, *conclusion* ou *attribut cible*.

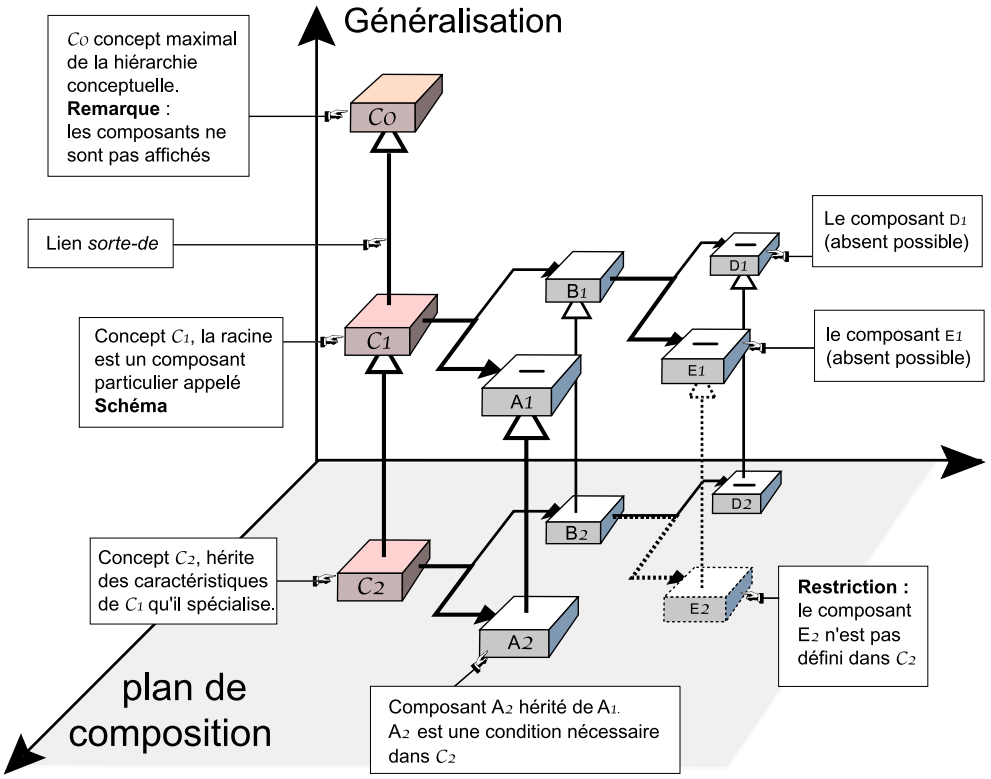


FIG. 2.10 – *Organisation verticale et horizontale des concepts dans CoDesc.*

d'ordre partiel (attribut hiérarchique). Dans ce cas, la hiérarchie des classes est exprimée par la hiérarchie des valeurs définissant le domaine de l'attribut (fig. 2.11).

Dans cette définition, les classes ne spécifient pas de contraintes particulières sur les individus. Ceux-ci sont instances d'un concept et possède une valeur particulière (éventuellement inconnue) qui définit leur classe d'appartenance. De plus, un individu peut appartenir à une ou plusieurs classes selon que la valeur de l'attribut de classe est simple ou multiple.

Le fait que l'attribut de classe appartienne à la liste des descripteurs du modèle induit une propriété intéressante, celle de permettre à l'utilisateur de la base de connaissances de changer l'attribut de classe, puis d'effectuer une synthèse des classes à partir des descriptions individuelles de la base et d'avoir ainsi un point de vue différent sur les instances, selon une classification différente.

Par exemple, l'utilisateur peut s'intéresser à la *forme générale* des individus décrits. En regroupant les individus en fonction de la *forme générale*, il obtient une caractérisation des individus pour chaque type de formes présentes dans les observations. Par défaut il obtient une caractérisation de chacune des classes définies dans le domaine de l'attribut taxon (cf.§ 3.5.4).

2.8 Les cas, descriptions des individus

Les objets élémentaires de notre étude sont des entités *observées* que l'on nomme des **individus**, qui doit être pris au sens de :

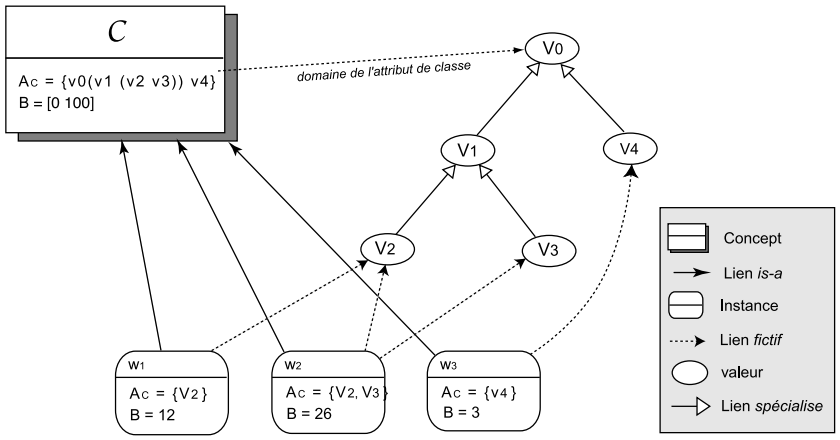


FIG. 2.11 – *Exemple de relations entre les individus d'un concept et la taxonomie de classes. L'attribut de classe A_C est de type hiérarchique. Les individus w_1, w_2 et w_3 sont instances du concept $\mathcal{C}$. Ils sont également membres d'une hiérarchie (implicite) de classes isomorphes au domaine de l'attribut de classe A_C.*

*"**Individu** : Terme inférieur d'une série, qui ne désigne plus de concept général et ne comporte plus de division logique." (extrait du dictionnaire Petit Robert)*

En biologie, un individu correspond à un **organisme** ou un **spécimen** du domaine d'étude. Par exemple, dans le cadre de l'application *base de connaissances sur les coraux des Mascareignes*, les individus étudiés sont des colonies de coraux, référencées dans des collections. Un individu désigne dans ce cas un ensemble d'organismes élémentaires (les *polypes*) vivants dans des *calices* et regroupés au sein d'une colonie. L'étude d'un tel individu ne procède pas d'une division logique supplémentaire, à savoir l'étude de chacun des calices [56] constituant la colonie. Ce qui est conforme à la définition. Cependant, le fait que l'on emploie le terme individu pour désigner une colonie ne veut pas dire qu'il soit impossible de la décomposer en un ensemble de constituants (les caractères de la description) mais indique simplement que ce niveau d'organisation hiérarchique est le plus détaillé auquel on s'intéresse.

Dans notre modèle, un **cas** (fig. 2.12) est le résultat d'un travail de description d'un individu. Il est dit instance d'un concept. Les cas sont regroupés au sein d'une (ou plusieurs) *base de cas*. Un cas hérite donc de toutes les caractéristiques définies par le modèle descriptif, au moment de l'instanciation. Lors de la description, il est possible de mentionner explicitement l'absence de certains composants, contingents, et donc des composants et attributs dépendants, par *absence déduite*.

Les cas possèdent un numéro qui permet de les référencer de manière unique, ainsi qu'un certain nombre d'attributs utiles pour la gestion de la base comme le descripteur [57], la date de création, etc. regroupés au niveau du composant *identification*. Un cas peut être associé à une ou plusieurs classes, ce qui est indiqué par la valeur de l'attribut de classe (le taxon).

Notons que dans l'état actuel que le modèle ne peut considérer qu'un seul attribut à la fois

56. L'observation des polypes étant impossible du fait que les objets étudiés sont des squelettes de colonies, on s'intéresse à leur habitat, les calices. Les parties molles (organiques) ont disparu du spécimen en collection.

57. l'auteur de la description

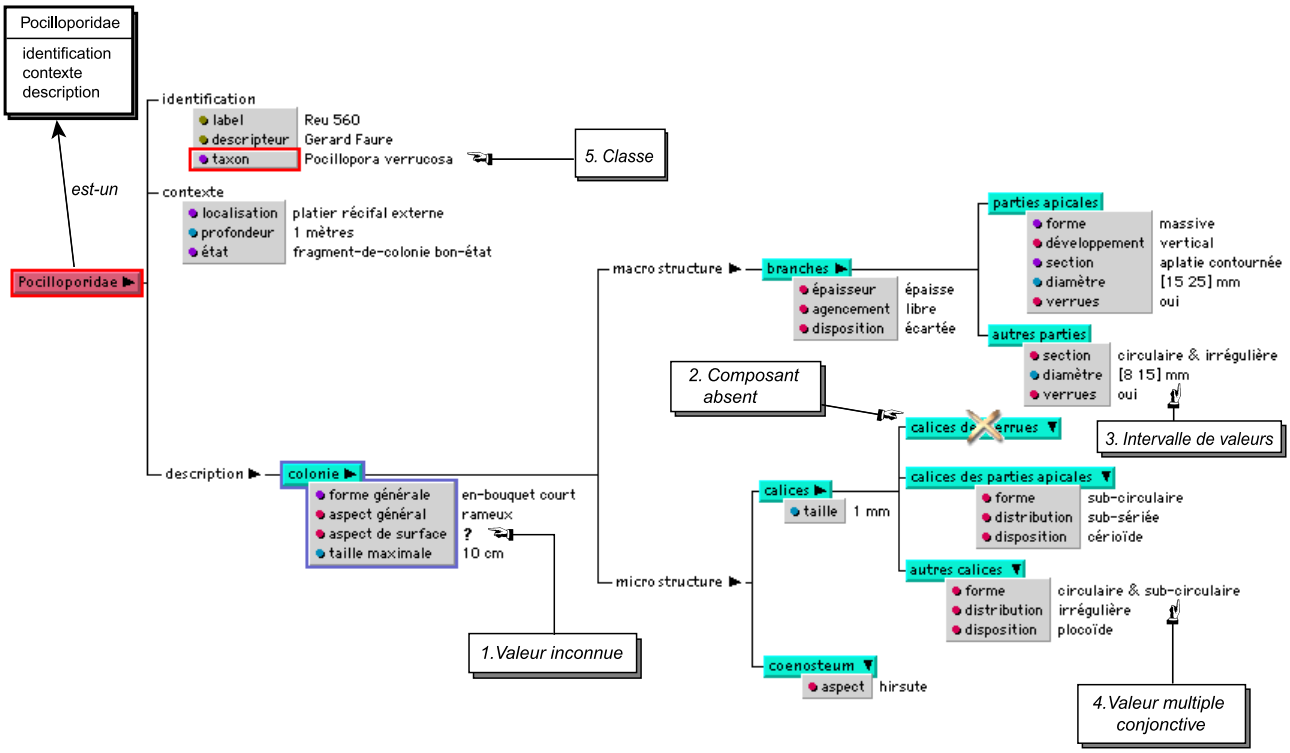


FIG. 2.12 – *Exemple de cas du modèle CoDesc dans IKBS. La structure du cas est identique à celle du modèle descriptif dénotant le concept* ***Pocilloporidae*** *(fig. 2.1).*

pour caractériser la classe, et non pas un ensemble de variables, comme c'est le cas généralement dans les modèles de raisonnement à partir de cas (RàPC) (Auriol 1995).

La figure 2.12 donne l'aperçu de la structure globale d'un cas. L'exemple est issu de la base de connaissances sur les coraux des Mascareignes. Les caractères essentiels de la description sont des composants du point de vue *description*. La classe d'appartenance d'un attribut est donnée par la valeur associée à l'attribut de classe *Taxon*. Notons que le composant *calices des verrues* est absent de la description, ce qui est conforme à la définition du modèle, étant donné que ce composant est *absent possible* (cf. figure 2.1). Le cas présenté est une description de l'échantillon n° **Reu 560**, décrit par **Gérard Faure** et appartenant à l'espèce *Pocillopora verrucosa*. Certains composants peuvent être absents, comme (2) **calices des verrues**. (1) désigne une valeur inconnue, (3) un intervalle de valeurs numériques observées simultanément, (4) un ensemble de valeurs symboliques observées simultanément.

La figure 2.13 illustre la relation *est-un* qui lie une instance à un concept, selon une perspective définie par le plan de composition et le plan de généralisation. L'individu **I** est instance du concept C_1. Tous les composants définis dans C_1 sont par défaut présents dans la description de **I**, sauf ceux qui sont explicitement notés *absent* comme par exemple le composant E_1.

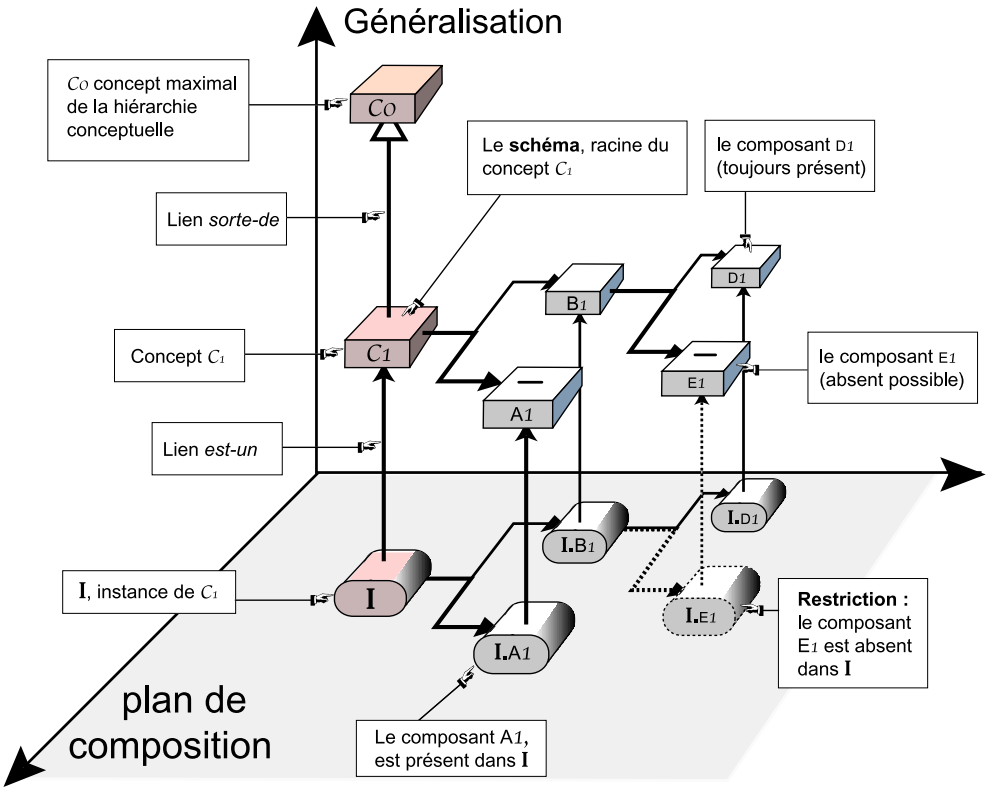


FIG. 2.13 – *Instanciation dans CoDesc.*

2.9 Les valeurs

Comme nous l'avons vu, la structure d'un cas est « héritée » du modèle descriptif dénotant le concept auquel il est attaché. Elle est donc contrainte par la définition du modèle et ne peut être modifiée au niveau des cas eux-mêmes. Cependant la présence de composants de multiplicité supérieure à 1 autorise les cas à préciser un nombre d'occurrences pour de tels composants, ce qui induit des variations locales entre les descriptions :

- Par l'« élagage » d'une branche de l'arbre provoqué par l'introduction de la valeur *absente* pour un composant absent-possible,
- Par l'instanciation de plusieurs occurrences d'un composant de multiplicité supérieure à 1.

Il est possible d'associer aux composants absents possibles et à chaque occurrence des composants de multiplicité supérieure ou égale à 1, un pseudo-attribut booléen traduisant la présence/absence de l'occurrence. Ainsi la description d'un cas peut être traduite par l'ensemble des valeurs observées pour chaque attribut et la donnée des pseudo-attributs. On effectue ainsi une correspondance d'une description structurée vers une représentation vectorielle des descriptions. La donnée du modèle descriptif et du vecteur de valeurs permet de reconstruire la structure d'un cas. Une base de cas constitue ainsi une représentation tabulaire *attribut-valeur* mixte, comme en analyse symbolique des données (cf. 1.9.5).

Concernant les valeurs présentes dans ce tableau mixte, la définition des attributs au niveau du modèle descriptif ne précise pas si une valeur prise par une observation est de type simple

ou multiple. Ce qui est le cas dans les SRCOs, pour lesquels des facettes de typage viennent qualifier la nature des valeurs que peuvent prendrent les attributs : type *liste* (ensemble de valeurs ordonnées) ou type *ensemble* (non-ordonné) par exemple.

En effet, on définit formellement un attribut comme étant une application à valeur dans l'ensemble des parties de son domaine :

Définition 2.4 *Soit $A \in \mathcal{A} = \{A_1, \ldots, A_i, \ldots, A_p\}$, un descripteur appartenant à l'ensemble fini de tous les descripteurs (composants et attributs) d'un concept $\mathcal{C}$.*

Pour toute observation, A est considéré comme une ***application*** *de Ω dans l'ensemble des parties de son domaine :*

$$\begin{aligned} A : &\Omega \rightarrow \mathcal{P}(\boldsymbol{dom}(A)) \\ &\omega \mapsto A(\omega) = v_\omega^A \end{aligned}$$

où v_ω^A désigne la valeur de type ensemble prise par ω pour le descripteur A.

Ainsi, l'espace de description des cas est :

$$\mathcal{V} = \prod_{i=1}^{i=p} v^{A_i}$$

Définition 2.5 *Un cas $\omega \in \Omega$ est défini par le p-uplet :*

$$v_\omega = (A_1(\omega), \ldots, A_p(\omega))$$

Egalement noté :

$$\omega = v_\omega^{A_1} \times \ldots \times v_\omega^{A_p}$$

Différents types de valeurs peuvent être interprétés de cette définition, selon que la cardinalité de la valeur v_ω^A est 0, 1 ou un ensemble de valeurs simples de $\mathbf{dom}(A)$.

2.9.1 Les types de valeurs

Valeur simple

Une valeur simple (ou atomique) peut être un entier, un réel, un intervalle de valeurs ou un symbole. Une valeur simple est un élément de $\mathbf{dom}(A)$ et donc de $\mathcal{P}(\mathbf{dom}(A))$. Nous notons abusivement les valeurs simples v, plutôt que par le singleton $\{v\}$. Par exemple, pour l'attribut continu $A \subseteq [0\ 100]$, une valeur simple peut être une valeur discrète (par exemple 10) ou un intervalle de valeurs inclus dans [0 100], comme [5 10] ou]10 20[.

Valeur de type ensemble

Une valeur de type ensemble $v = A(\omega)$ est un sous-ensemble $\mathcal{P}(\mathbf{dom}(A))$ qui est interprété comme une **disjonction** des valeurs simples contenues dans v. Par exemple, soit A un attribut symbolique de domaine $\mathbf{dom}(A) = \{bleu, rouge, jaune\}$ (cf. fig. 2.8). La valeur ensemble $v_\omega^A = \{bleu, jaune\}$ est interprétée comme la disjonction $bleu \vee jaune$.

Pour un attribut numérique de domaine $\mathbf{dom}(A) \subseteq [0\ 100]$, une valeur multiple correspond à un ensemble de valeurs simples disjointes, discrètes ou intervalles, par exemple $\{3, [5\ 10],]15\ 30], 50\}$. Notons qu'une fonction de normalisation vérifie la cohérence de ce type de valeurs, afin de garantir que les valeurs contenues dans l'ensemble sont disjointes, bornées par le domaine de définition et ordonnées (par l'ordre naturel sur les éléments de $\mathbb{R}$ ou de $\mathbb{N}$). Nous reviendrons au chap. 3 sur les opérations possibles sur les valeurs en fonction de leur type.

Valeur conjonctive

Alors qu'une valeur de type ensemble correspond à une disjonction des valeurs simples contenues, ce qui permet d'exprimer l'**imprécision** de l'observation, les valeurs conjonctives, notées par exemple $bleu \wedge jaune$, traduisent la **variabilité** du caractère correspondant dans les observations. Ce type de valeurs est donc utilisé lorsque le descripteur observe simultanément plusieurs valeurs différentes.

Le principe de choix entre l'expression d'une conjonction ou d'une disjonction est donc :

« Une conjonction de valeurs exprime une variation, une disjonction exprime une imprécision »

La sémantique étant donnée, les conjonctions posent cependant un problème d'interprétation, notamment lors du traitement. En effet, quelle relation existe-t-il entre les valeurs simples du domaine de l'attribut et une conjonction de valeurs ? En d'autres termes, entre $v = bleu \wedge jaune$ et la couleur *bleu* ? Peut-on dire que *bleu* est plus général ou plus spécifique que v ?

Nous avons retenu essentiellement trois interprétations possibles :

- La première interprétation est de considérer qu'une valeur conjonctive est une **nouvelle valeur**, indépendante des valeurs simples qui la composent. Cette interprétation amène également à considérer que le domaine de l'attribut est implicitement augmenté de toute combinaison des valeurs simples déclarées. Ce qui a pour effet d'augmenter rapidement le domaine de l'attribut. $bleu \wedge jaune$ est donc une nouvelle valeur, indépendante du *bleu* et du *jaune*.
- Dans la seconde interprétation, la valeur $bleu \wedge jaune$ est plus spécifique que *bleu*. Le domaine d'un attribut nominal est alors une hiérarchie de valeurs, les valeurs les plus générales étant celles déclarées explicitement par le concepteur. Ainsi, $bleu \wedge jaune$ est une couleur un peu particulière, qui tend vers le *vert* ...
- La troisième interprétation considère que $bleu \wedge jaune$ est équivalent $\{bleu, jaune\}$, c'est-à-dire à une disjonction. Cette interprétation peut paraître sémantiquement incorrecte. Pas nécessairement. Devons-nous effectuer une distinction entre une variation observée et une imprécision qui masque généralement une variation délicate à mettre en évidence ? La réponse des experts va généralement dans ce sens. Ainsi, la distinction imprécision / variation est effective au niveau de la représentation des connaissances, mais aucune distinction n'est faite lors du traitement par les méthodes d'analyse.

Les algorithmes que nous avons développés proposent lors du paramétrage de la méthode à appliquer, ces différents choix d'interprétation des conjonctions de valeurs. Nous discutons et

justifions, en conclusion de ce chapitre, de la pertinence du modèle CoDesc et des choix que nous avons effectués pour l'expression de la variabilité et de la diversité de l'Observé.

Valeur vide

Une valeur vide (ou ensemble vide) est un cas particulier de valeur simple pour lequel aucune occurrence d'élément de $\mathbf{dom}(A)$ n'est précisée. Cette valeur est noté $\{\}$ ou encore $\emptyset$. Plusieurs interprétations sont possibles pour l'ensemble vide :

- valeur **non-applicable**. Elle est comparable aux valeurs notées **N-A** utilisées en analyse de données pour signifier qu'une valeur est impossible à observer, du fait d'une relation de dépendance de type *mère-fille* (Brito 1991, Lebbe 1991) entre descripteurs, ce qui est le cas dans le modèle CoDesc pour les descripteurs contingents dont l'absence est déduite par l'absence d'un ancêtre.
- valeur **non-renseignée**. Lorsqu'un nouveau cas vient d'être créé, toute les valeurs sont vides, sauf dans le cas où une valeur par défaut à été explicitement renseignée lors de la définition de l'attribut. La totalité des descripteurs nécessaires doit être renseignée pour que le cas soit « valide » et puisse appartenir au concept, ce qui est vérifié par le système à l'aide d'une fonction booléenne qui teste la validité des cas.
- valeur **calculée**. Lors de la synthèse d'un nouveau cas ω', par intersection des valeurs d'un ensemble de cas $\{\omega_1, \ldots, \omega_i, \ldots, \omega_n\} \in \Omega^n$ d'un concept $\mathcal{C}$, certaines valeurs ω' peuvent être vides.

Valeur inconnue

Lorsqu'un descripteur ne peut être observé, la valeur inconnue est utilisée. Elle est plus générale que tout autre valeur et formellement équivalente au domaine de l'attribut. On la note * ou ?.

Valeur exceptionnelle

Une valeur exceptionnelle est une valeur renseignée dans une description qui n'appartient pas au domaine de définition de l'attribut. Elles sont possibles pour tout type d'attribut et sont très utiles, en particulier lorsque la définition précise de modèle descriptif n'est pas achevée. Pour un attribut numérique, est considérée exceptionnelle une valeur (discrète ou intervalle) supérieure ou inférieure aux bornes de l'intervalle de définition. Pour un attribut symbolique, c'est un symbole non renseigné dans le domaine de définition. Pour un attribut hiérarchique, elle peut être un nouveau symbole, ou bien un nouveau symbole spécialisé d'un symbole existant. Dans l'exemple 2.7, l'utilisateur peut renseigner la valeur claire *bleu* qui est une valeur non prévue, mais cependant plus précise que *clair* qui est définie dans le domaine de l'attribut *couleur*.

Soit v une valeur exceptionnelle, $A \in \mathcal{A}$ et $\tau(A)$ un attribut et $\tau(A)$ le type associé. Les différentes méthodes d'analyse interprètent les valeurs exceptionnelles en fonction du type de A de la façon suivante :

- $\tau(A)$ = "ordinal" ou "nominal" : une valeur exceptionnelle est traitée comme la valeur

inconnue.
- $\tau(A)$ = "numérique" : la restriction de v au domaine de A est considéré par intersection, soit $\mathbf{dom}(A) \cap v$.
- $\tau(A)$ = "taxonomique" (hiérarchique) : la valeur la plus spécifique du domaine modélisé présente dans v est considérée. Soit *clair* dans l'exemple précédent.

Les valeurs exceptionnelles ne sont donc pas traitées comme une information utilisable par les méthodes d'analyse, puisqu'elles ne sont pas validées au niveau du modèle par l'expert. Cependant, l'information qui est utilisée est la restriction (intersection) de la valeur exceptionnelle au domaine de l'attribut. De plus, ces valeurs sont conservées dans les cas et permettront par la suite d'étendre le domaine des attributs, lors de la révision du modèle descriptif.

2.9.2 Treillis sur les co-domaines

Il est toujours possible de munir le co-domaine[58] d'un attribut d'une structure de sup-demi treillis (fig. 2.14), en prenant comme relation d'ordre l'inclusion et comme opérateur sup, l'union. Cet ordre peut être vu comme une relation de généralisation sur le co-domaine. On peut aller plus loin et définir sur chaque co-domaine, une structure de treillis où la relation d'ordre est alors interprétée comme une relation de généralisation/spécialisation (cf. (Girard 1997)). Le plus petit élément du treillis est la valeur inconnue $* = \cup\{d \in \mathcal{P}(\mathbf{dom}(a)\}$ et le plus grand élément est l'ensemble vide $\emptyset$.

Proposition 2.1 *Chaque co-domaine* $\boldsymbol{cod}(A)$ *d'un attribut symbolique A est muni d'une relation d'ordre et d'une structure de treillis (*$\boldsymbol{cod}(A), \leqslant, \vee, \wedge, *, \emptyset$*) en lui adjoignant un élément maximal* $*$ *et un élément minimal* $\emptyset$*. La relation d'ordre est obtenue en inversant l'ordre induit par l'opérateur d'inclusion.*

Soit $a \in \mathcal{A}$*, un attribut symbolique :*

$$\forall x,y \in \boldsymbol{cod}(A) = \mathcal{P}(\boldsymbol{dom}(A)) : x \leqslant y \Longleftrightarrow x \subseteq y$$

y est plus générale que x.

La figure 2.14 montre les treillis construits à partir d'un domaine de type nominal $\mathbf{dom}(A) = \{bleu, rouge, jaune\}$ et d'un domaine de type taxonomique.

2.10 Les données contextuelles

Les **données contextuelles** sont constituées d'une part, de connaissances non formalisées, mais utiles au domaine, telles que des textes de référence du domaine, des monographies, des publications scientifiques. D'autre part, elles servent à apporter des renseignements utiles pour mieux comprendre et interpréter les concepts définis au sein de la base de connaissances.

La plate-forme $\mathcal{IKBS}$ (cf. chap. 6), dans laquelle est implantée le modèle CoDesc, permet de gérer différents formats de documents : formats texte standard : (ASCII, RTF, PDF), des

58. **Co-domaine** d'un attribut : ensemble des valeurs pouvant être renseignées dans les descriptions sans considérer les valeurs exceptionnelles.

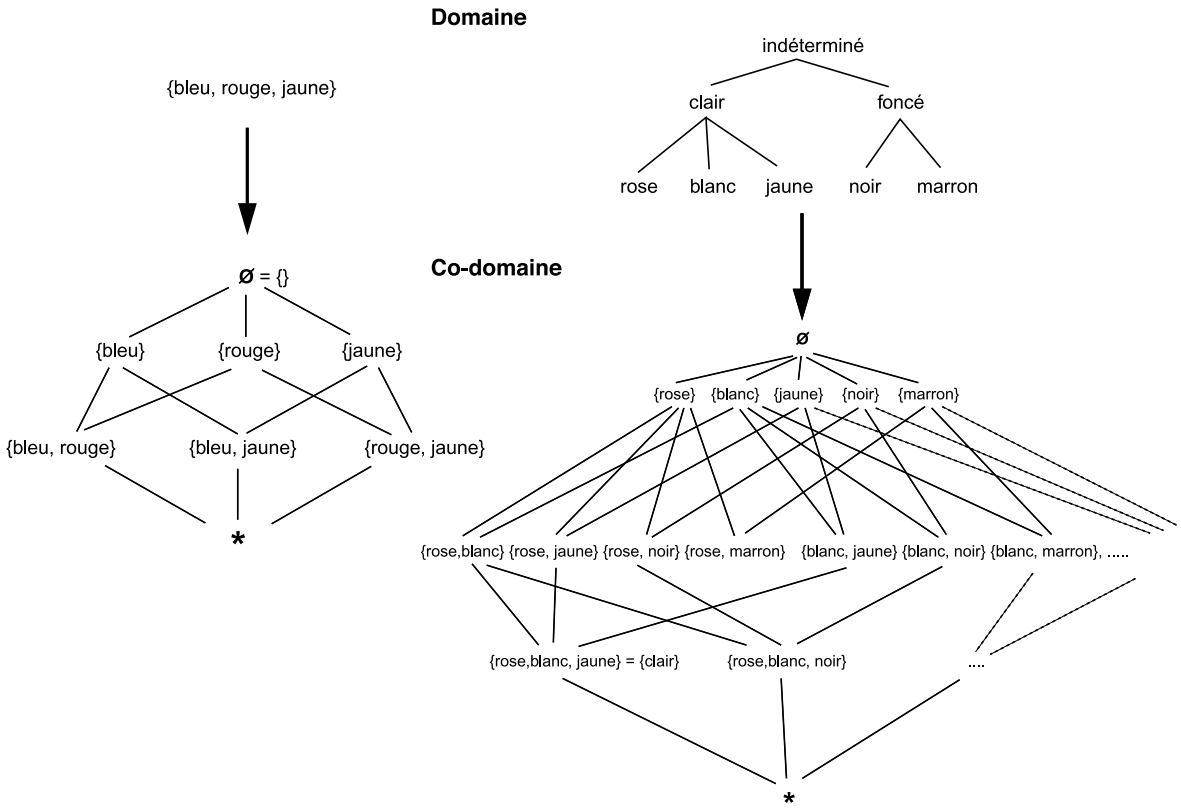


FIG. 2.14 – *Exemple de structures de treillis sur les co-domaines d'un attribut nominal (à gauche) et taxonomique (à droite).*

liens hypertextes (URL), des pages HTML, des images (GIF, JPEG, TIFF), des formats vidéo (Quicktime, MPEG, AVI). Certains sont gérés directement par le système (fig. 2.15), l'affichage des autres est délégué à l'application externe correspondante (navigateur internet, éditeur de texte, etc.).

En particulier, la base de connaissances sur « les coraux des Mascareignes » contient un grand nombre d'illustrations explicatives sur la manière d'interpréter correctement les caractères. Voir l'annexe B pour un aperçu de quelques illustrations du modèle *Pocilloporidae*.

Ces différentes ressources sont reliées aux éléments constitutifs de la base de connaissances, par l'intermédiaire de relations de type *référence*. Dans la définition d'un modèle descriptif, elles peuvent être attachées aux composants, aux attributs et aux valeurs définies dans chaque domaine. Dans la description d'un cas, elles peuvent être associées aux valeurs renseignées.

- Associées au *Schéma* : informations d'ordre générales sur le concept modélisé, les différents taxons qu'il recouvre, glossaire de la terminologie employée, etc.
- Associées aux composants et aux attributs : informations concernant la manière d'interpréter correctement les caractères et renseigner les valeurs. Le module d'illustrations permet d'associer des illustrations aux valeurs définies dans les domaines des caractères (voir fig. 2.15). Ceci est très utile pour identifier un nouveau cas par photo-interprétation.
- Associées aux valeurs des cas : permet d'illustrer les spécimens d'une collection et leur

spécificité. Ces illustrations (photos) et les valeurs associées peuvent être « remontées » automatiquement au niveau du modèle, afin de renseigner directement les caractères définis au sein du modèle.

Des annotations et des mots clefs peuvent être associés à ces données contextuelles (fig. 2.15) pour expliciter, commenter le contenu, etc. Les mots clefs permettent d'effectuer une recherche dans l'ensemble des données contextuelles présentes dans la base de connaissances.

FIG. 2.15 – *Exemple de données contextuelles associées à un descripteur du modèle descriptif, l'attribut localisation du composant Contexte. Des commentaires et des mots clefs peuvent être associés à chaque descripteur.*

2.11 Elaboration d'une base de connaissances avec CoDesc

Après cet aperçu du modèle conceptuel de représentation des connaissances CoDesc, on s'intéresse dans cette partie à quelques aspects pouvant guider les experts dans l'élaboration d'une base de connaissances. Les détails permettant d'*améliorer la robustesse des systèmes d'aide à la description, à la classification et à la détermination des objets biologiques* sont développés dans Conruyt (1994).

2.11.1 Acquisition de l'Observable

L'acquisition de ce qui est potentiellement observable consiste dans un premier temps à définir un modèle descriptif (concept) pour un sous-domaine du domaine considéré, et donc à identifier au préalable ce sous-domaine. Les critères pouvant guider ce choix peuvent être par exemple le degré d'*homogénéité* des observations relevant du concept ou encore la réalité taxonomique. On s'intéressera à une famille, un genre ou une espèce particulière.

La conception d'un modèle est ensuite fondée sur l'observation de la variabilité de l'Observé. Ceci est un travail de spécialiste, car il nécessite une bonne expérience et une certaine familiarité

avec le domaine. En effet, caractériser d'emblée tout ce qui est observable est une tâche très difficile, qui requiert une vision synthétique et complète du domaine. Il en résulte que le choix des objets composants, des différents attributs, valeurs et règles mis en jeux, ne peuvent en pratique être réalisés dès le départ. La démarche que nous proposons, dite *itérative* ou incrémentale permet d'affiner progressivement le modèle initiale (premier jet) par modifications successives, sur la base des descriptions (l'observé) de spécimens décrits à l'aide du modèle. Grâce à ces descriptions organisées dans une base de cas, les méthodes d'analyse peuvent également aider l'expert à reconsidérer ses choix : ajouter de nouveaux descripteurs, modifier les noms, etc. lors de l'étape de validation.

2.11.2 Les objectifs de la modélisation

Définir les éléments d'un modèle descriptif nécessite de caractériser clairement les objectifs de la modélisation : représenter de « bonnes » descriptions ne suffit pas, il faut au préalable définir clairement le but de la modélisation. Hormis l'objectif intrinsèque de l'informatisation des données de collection, nous distinguons essentiellement quatre objectifs pouvant guider les choix des éléments intervenant dans un modèle descriptif :

1. inférence de taxonomie, à partir de descriptions,
2. classification d'objets,
3. identification d'individus,
4. détermination d'objets.

Inférence de taxonomies

C'est un des objectifs majeur pour les biologistes. L'inférence de taxonomies consiste à trouver une taxonomie qui organise un ensemble d'individus. Elle est appelée classification en analyse de données et classification conceptuelle en apprentissage. Pour les biologistes, la classification d'individus est une démarche exploratoire qui vise à remettre en cause les classifications préexistantes. Les descriptions de spécimens constituent la base de ce travail, qui consiste à mettre en oeuvre un ensemble de méthodes de classification comme par exemple des classifications ascendantes hiérarchiques (CAH), puis étudier les structures obtenues, en comparaison avec les classifications préétablies.

L'objectif de la modélisation est alors de saisir le maximum de caractères sur les individus, sans *a priori* sur leur rôle et leur utilité pour une classification : l'**exhaustivité des descriptions** d'individus est recherchée (Conruyt 1994).

Classification d'objets

La **classification d'objets** (ou de caractères) focalise l'attention sur un objet particulier (un composant) du modèle. L'expert vise à déterminer le rôle de l'objet dans la classification, en particulier s'il correspond à un caractère diagnostique, c'est-à-dire qui permet de séparer de façon non-ambiguë les taxons. On cherchera alors à déterminer s'il s'agit d'un caractère stable, indépendant de facteurs extérieurs tels que le milieu écologique par exemple. D'une façon générale,

la classification d'objets vise à améliorer la connaissance taxonomique d'un groupe particulier. La taxonomie est en effet une théorie très complexe qui émerge de l'étude des caractères, de leur variabilité, de leur signification phylogénétique ou écologique. Dans un grand nombre de groupes zoologiques, la connaissance incomplète ne permet pas de construire une théorie définitive. C'est le cas par exemple des éponges marines ou des coraux.

Pratiquement, l'expert cherchera à étoffer la description de l'objet à classifier par un nombre important de caractères propres. Cet objectif correspond à un besoin d'homogénéisation du vocabulaire dans la communauté des chercheurs du domaine ainsi que la recherche d'homologies entre caractères.

Identification d'individus

Appelée identification en analyse de données, classement ou classification (d'instances) en représentation par objets. Elle consiste à déterminer une ou plusieurs classes auxquelles un individu peut appartenir.

L'identification d'individus consiste pour les biologistes à associer la description d'un spécimen observé à un taxon. D'un point de vue informatique, elle vise à identifier un ensemble d'individus dans une hiérarchie de concepts (ou classes) préexistante. La détermination n'est pas nécessairement une affaire de spécialiste, elle peut être mise en oeuvre par des non-spécialistes du domaine, à l'aide d'outils adaptés. Nous proposons par la suite (chap. 4) une méthode d'identification assistée par ordinateur (I.A.O.) travaillant à partir des descriptions de CoDesc.

Cependant, ces utilisateurs n'ont pas toujours une connaissance suffisante du domaine, ni les outils (loupe binoculaire, microscope, etc.) nécessaires pour observer correctement les caractères du spécimen à identifier. La **redondance des caractères** ne doit alors pas être écartée lors de la conception du modèle descriptif, afin que les corrélations entre caractères permettent de remplacer ceux auxquels l'utilisateur ne sait pas répondre (caractères alternatifs). D'autre part, les poids associés aux caractères par le concepteur du modèle permettent de définir dans cet objectif, un préordre sur les caractères en fonction de leur **facilité d'observation**.

Détermination d'objets

Elle consiste à déterminer lors de la description d'un spécimen de l'étude, pour un composant polymorphe particulier, le sous-composant le plus approprié dans la hiérarchie locale de composants (cf. spécialisation des composants). Elle permet de renseigner de manière automatique certains objets d'une description, en fonction des valeurs des caractères de l'objet générique renseigné par le descripteur. Elle apparaît donc sous ce point de vue comme une assistance à l'observation. Nous n'avons pas développé à ce jour d'algorithme spécifique de détermination d'objets, qui soit localisé à une hiérarchie de composants. Cet aspect apparaît comme un axe de recherche qui, conjointement aux fonctionnalités hypertextuelles, multimédia, et picturales, concerne l'assistance à l'utilisateur pour la description d'objets complexes.

Dans cet objectif, l'expert doit décrire de manière fine et détaillée certains objets, élaborer des hiérarchies locales (spécialisation) en introduisant éventuellement la possibilité de décrire plusieurs fois les mêmes composants (multiplicité).

2.11.3 Discussion sur la modélisation

Dans un objectif de classification et d'identification d'individus, l'exhaustivité et la redondance des informations qui permettent de former un modèle complet est importante. L'expert doit être capable de synthétiser le maximum de connaissances du domaine, à partir des échantillons en collection (en particulier les types) et des monographies, pour constituer un modèle descriptif qui définit l'*Observable*.

L'expert doit aussi interpréter l'*Observé* existant en dégageant les faits marquants d'une bonne description (Conruyt 1994). Les plus importants sont réifiés en composants, détaillés par un ensemble d'attributs de différents types. L'objet principal est le schéma, racine de l'arbre de description (ou de décomposition), généralement nommé directement par le nom qui désigne au mieux le concept modélisé. Les objets sont ensuite mis en relation, en fonction de la nature des relations qu'ils entretiennent (composition ou spécialisation), de leur niveau de généralité (du plus général au plus particulier).

Nous donnons un exemple d'application issu de la « base de connaissances sur les coraux des Mascareignes », pour illustrer la difficulté de modéliser certains objets un peu particulier.

2.11.4 Exemple d'application : l'objet septes du modèle *Pocilloporidae*

Le cas d'étude porte sur la modélisation des **septes** des calices de la famille des *Pocilloporidae* (fig. 2.16), plus particulièrement sur leur distribution.

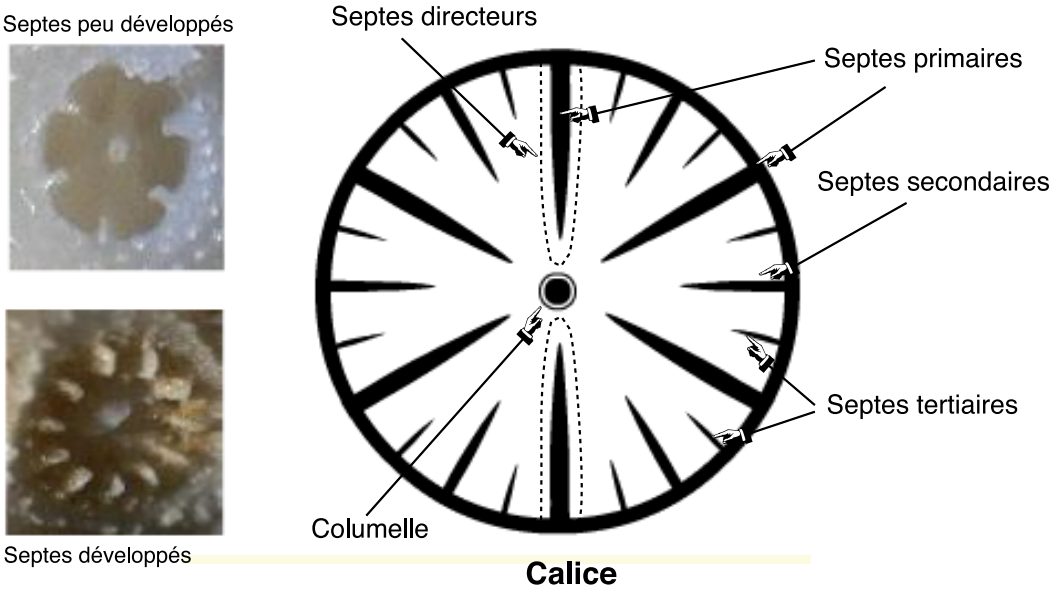


FIG. 2.16 – *Illustration de la distribution en cycles des* ***septes*** *des calices de la famille des Pocilloporidae. Les septes directeurs sont des septes primaires particulièrement développés, qui parfois se soudent à la columelle.*

Définition 2.6 *Septes (Scléroseptes)*

Lames calcaires verticales disposées selon une symétrie radiaire d'ordre 6 (Hexacoralliaires), et qui divisent la cavité calicinale en chambres (multiple de 6).

Commentaire de l'expert concernant la distribution des septes

Les septes sont distribués en un, deux, ou trois cycles, dont souvent, seuls les deux premiers sont bien visibles. Il y a au total 12 septes. Le premier cycle renferme 6 septes (septes primaires ou de premier ordre). Le second cycle renferme également 6 septes (septes secondaires ou de second ordre). Le troisième cycle contient 12 septes (septes tertiaires ou de troisième ordre). Cela donne un total maximum de 24 septes. En pratique, il arrive que des cycles soient incomplets.

Les septes primaires sont généralement les septes les plus développés. Cependant, parfois, les septes de deuxième ordre présentent un développement égal ou sub-égal à ceux du premier ordre.

Dans certains cas, un ou deux septes du premier cycle sont plus développés que les autres : ils constituent les septes directeurs et marquent également l'axe dorso-ventral du calice. Ces derniers peuvent par ailleurs se souder avec la columelle constituant avec cette dernière une cloison continue séparant le calice en deux parties égales. Dans d'autres cas, c'est l'ensemble des septes de premier ordre qui rejoint la columelle. Cette soudure peut s'établir à différents niveaux, depuis le plancher du calice jusqu'à la partie sommitale de la columelle.

Les septes tertiaires sont généralement peu développés, voire abortifs (matérialisés le plus souvent par des rangées discontinues d'épines verticales intra-calicinales). Dans certains cas, ce dernier type de développement peut également se rapporter aux autres types de septes (indépendemment des ordres auquels ils appartiennent). L'identification des septes, leur dénombrement ainsi que leur appartenance en cycles s'avère alors difficile.

Modélisation

La modélisation de ce type d'objet complexe a suscité de nombreuses discussions au sein de notre équipe. En effet, plusieurs modélisations sont possibles :

1. Utilisation d'une hiérarchie locale d'objets (fig. 2.17). Une conception possible consiste à modéliser un composant général, *septes*, dénotant l'ensemble des septes d'un calice, sans distinguer les types des sous-ensembles.
   Deux attributs sont attachés à l'objet : le *nombre* et la *distribution* correspondant au nombre de septes total et à la distribution de l'ensemble des septes.
   Le composant *septes-s*, de multiplicité [1 3], correspond à une sorte de septes non nommé [59]. *septes-s* est spécialisé en trois objets plus spécifiques correspondant respectivement aux *septes primaires*, *secondaires* et *tertiaires*. Notons que les *septes directeurs* sont une partie très développée des *septes primaires*, d'où la relation de composition entre les deux. Les septes directeurs peuvent être absents. A partir de cette modélisation, le descripteur a ensuite la possibilité de ne pas nommer le type de septes observés, selon ses facultés d'observation et de compréhension du domaine, en instanciant plusieurs fois *septes-s* sans les nommer. En fonction des valeurs renseignées, le système est capable de préciser automatiquement (détermination d'objet) le type des objets décrits. En particulier, si la propriété

59. Il joue le même rôle qu'une classe abstraite dans le modèle objet, à la différence qu'il peut être instancié.

soudure à la columelle est *vrai*, l'objet est de type *septes primaires*, et le composant *septes directeurs* est présent.

2. Modélisation plus « classique ». Un composant *septes* et trois sous-composants, correspondant respectivement aux quatre sortes de septes.

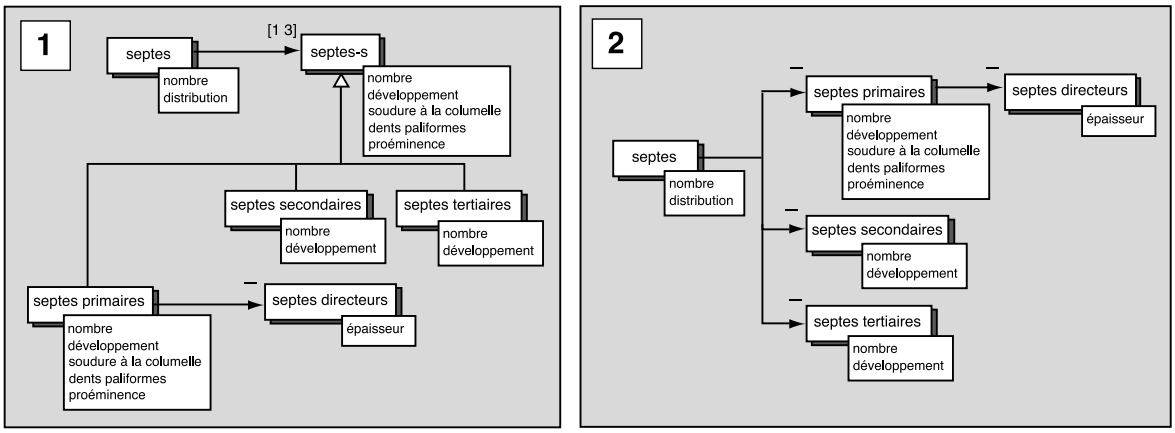


FIG. 2.17 – *Deux modélisations possibles de l'objet* ***septes****. **(1)** fait usage de la spécialisation et de la multiplicité, **(2)** utilise uniquement la composition.*

La modélisation s'est portée sur le second choix (fig. 2.18), pour deux raisons. D'une part, il est impératif de comprendre le principe de distribution des septes pour pouvoir les observer correctement. En particulier, seuls les *septes primaires* peuvent être soudés à la columelle (partie centrale des calices). D'autre part, le nombre de septes n'est pas aléatoire, il procède d'une certaine logique, 6, 12 ou 24 septes peuvent être observés, selon que seuls les *septes primaires* sont présents, les primaires et les secondaires ou bien les trois types, respectivement. Il est donc important de ne pas confondre les trois types de septes, afin de ne pas compromettre cette « arithmétique de la nature ». Il est possible d'associer à l'attribut *nombre[septes]* une règle permettant de calculer automatiquement le nombre total de septes.

L'exemple 2.18 illustre une description locale des septes. Notons qu'il a été impossible pour l'auteur de décider (ou d'observer) la présence de *septes tertiaires*, qui sont *abortifs*, c'est-à-dire très peu développés voir presque inexistants. La valeur [0 12] associée au nombre de septes tertiaires caractérise donc une *imprécision*.

2.12 Conclusion et discussion

Nous avons présenté un modèle déclaratif de représentation des connaissances **descriptives** et **classificatoires**.

De l'exposé de ce modèle et des différentes interprétations, nous posons le problème général de l'expression de l'incertain, du possible, de la variation et du flou en représentation des connaissances. En particulier lorsqu'il s'agit d'exprimer la biodiversité.

Jusqu'à quel degré de précision devons-nous prendre en compte la subtilité des objets de la Nature? Devons nous quantifier plus précisément la variation et l'incertitude?

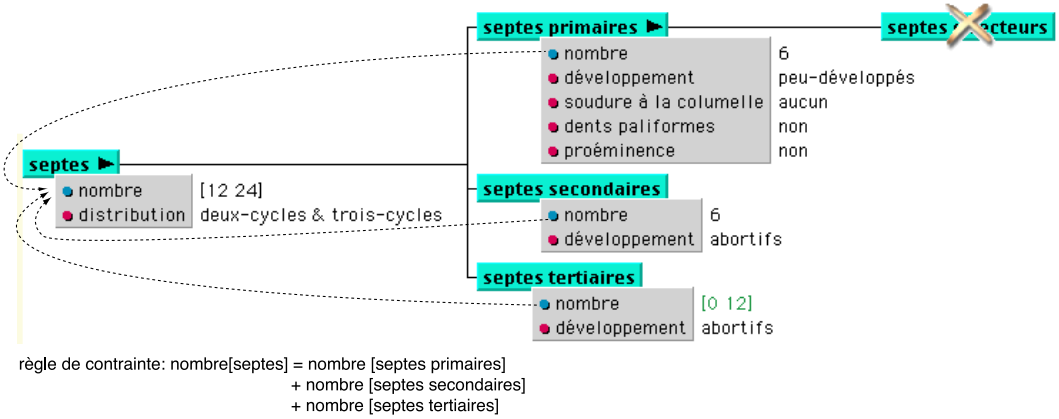


FIG. 2.18 – *Cas d'un échantillon présentant trois cycles distincts. Entre 12 et 24 septes sont observés, le descripteur ne pouvant décider si les septes tertiaires sont présents. Ce cas ne présente pas de septes directeurs (objet marqué d'une croix).*

Nous pourrions par exemple introduire des types d'attribut supplémentaires, pour exprimer la prédominance d'une valeur par rapport aux autres dans l'expression d'un fait conjonctif ou disjonctif, accorder de l'importance à l'ordre des valeurs, ou encore associer des nuances (à peu près, presque toujours, etc.) aux valeurs, sous la forme de treillis nuances [60]. Nous pourrions également développer un langage d'expression des relations, plus subtil, fondé sur une sémantique formelle bien définie, tel que dans les « logiques terminologiques » (§ 1.10.3).

La position des experts nous a beaucoup éclairée dans l'élaboration de ce modèle. Effectuer un diagnostic consiste à faire des choix, émettre des hypothèses, quitte à les remettre en cause ultérieurement. Parfois ces choix sont difficiles, mais il faut bien les faire à un moment ou à un autre ! En d'autres termes, la logique des propriétés nécessaires et contingentes que nous avons élaborée (l'absence possible de certaines parties de l'Observable et l'absence avérée de certains objets de l'Observé), ainsi que la logique des valeurs conjonctives et disjonctives, nous paraissent suffisantes pour exprimer la variation et l'imprécision des observations. Le modèle nous semble cohérent de ce point de vue.

En effet, affiner le pouvoir de représentation rend plus difficile et plus long le travail d'observation et d'acquisition des descriptions, ce qui amène inévitablement à acquérir moins de descriptions et donc à travailler sur un échantillonnage plus réduit. De plus, les algorithmes pouvant traiter des données plus complexes sont beaucoup plus subtils et délicats à mettre en oeuvre. Ce serait également nous priver du grand nombre de méthodes « plus classiques » issues de l'Apprentissage Automatique ou de l'Analyse de Donnée. Nous en avons adapté quelques unes pour l'identification et la classification des descriptions. Elles feront l'objet des chapitres suivants.

Nous pensons également que la spécificité du domaine de connaissances abordé, la systématique justifie nos choix. Les connaissances descriptives sont prises en compte à deux niveaux. Le modèle descriptif caractérise une catégorie d'individus, c'est-à-dire un taxon, et une instance décrit un spécimen unique (en collection). Ainsi, la variabilité de l'observé est prise en compte

60. Voir en particulier Girard (1997), qui représente les descriptions sous la forme d'« arborescences symboliques nuancées ».

lors du choix des spécimens à décrire, ainsi qu'au niveau de la sélection des descriptions. La définition d'une base de cas doit donc refléter, dans la mesure du possible, la distribution naturelle des organismes, et la variation des observations.

Ainsi, nous pensons qu'il est préférable de laisser le système inférer de nouvelles *descriptions synthétiques* à partir d'une base de cas bien construite - Celles-ci présentent l'intérêt de mettre en évidence des distributions de probabilité *a posteriori* - plutôt que de tenter de caractériser précisément et *a priori* des descriptions de concepts et les qualificatifs de fréquences (rares, très nombreuses, etc.) associés aux descripteurs, comme il est exprimé dans les monographies.

Les bases de cas ainsi constituées forment un *ensemble d'apprentissage* pour les méthodes d'analyse, qui permettent par exemple d'inférer de nouvelles *descriptions synthétiques*, d'étudier la variabilité des différents descripteurs pour un taxon donné et de construire de nouveaux concepts. Ce que nous illustrerons au chapitre suivant.

Enfin la spécificité de la Systématique des Scléractiniaires justifie la nécessité de représenter les connaissances à deux niveaux. La seule donnée d'une description de concept (un taxon) ne suffit pas dans ce domaine. En effet, les constituants biologiques des récifs coralliens sont de petits organismes, les *polypes*, qui forment des colonies de plusieurs milliers d'individus, au sein d'un milieu très changeant. Ainsi, la variabilité intra-spécifique, voir intra-coloniale, est très forte. Comment alors traduire cette variabilité sans la donnée des descriptions ? C'est bien une des raisons majeure pour laquelle les systématiciens du domaine ont bien des difficultés à s'accorder sur le nombre d'espèces de coraux vivants sur la planète (entre 300 et 700).

Repenser la Systématique, c'est repartir sur ses bases : les descriptions, émettre des hypothèses et tirer le meilleur parti de la capacité qu'ont les machines à calculer vite et juste, en leur communiquant les éléments nécessaires pour obtenir des résultats fiables.

Chapitre 3

Opérations internes du modèle CoDesc

Sommaire

3.1 Introduction

Dans la première partie de notre exposé du modèle CoDesc (chapitre 2), nous avons présenté l'aspect *statique*, c'est-à-dire la définition des différents constituants et des relations présents dans le modèle. Nous abordons dans ce chapitre l'aspect *dynamique* matérialisé par les opérations de bases possibles sur les éléments constitutifs du modèle.

Afin de faciliter la conception de méthodes d'analyse, et de conduire des raisonnements évolués sur les connaissances exprimées, chaque type d'élément (composants, attributs, valeurs, etc.) est muni d'un ensemble d'opérations de base. Ces opérations font intrinsèquement partie du modèle et permettent d'inférer de nouveaux éléments constitutifs. Nous présentons dans ce chapitre certaines d'entre elles, des plus simples aux plus complexes, qui permettent :

- D'effectuer des opérations sur les **valeurs**. Il s'agit d'opérations arithmétiques simples sur les descripteurs numériques, d'opérations ensemblistes (intersection, union, enveloppe convexe, etc.), d'opérations de généralisation (au sens du treillis de valeurs), etc.
- De rechercher le concept généralisant le plus spécifique d'un ensemble hétérogène composé d'individus et de concepts.
- D'inférer de nouvelles **descriptions** synthétique d'individus, à partir d'un ensemble de descriptions.
- De gérer l'*évolution* des connaissances, lorsque la nature des éléments constitutifs des concepts (modèles descriptifs) subissent des modifications.

Par ailleurs, le modèle possède d'autres opérations de base qui ne seront pas développées dans cete thèse, mais qui ont été implanté au sein de la plate-forme $\mathcal{IKBS}$. Ces opérations permettent :

- Mettre en évidence des **relations de généralisation** entre individus, entre individus et concepts, et entre concepts,
- De synthétiser des **concepts**. Synthèse de nouveaux concepts à partir de la donnée d'un ensemble de descriptions et des concepts associés.
- D'effectuer des opérations de **filtrage**, c'est-à-dire sélectionner un ensembles d'individus à l'aide d'un filtre exprimé par une assertion.

Nous donnons dans un premier temps un aperçu de la manière dont nous avons abordé la conception informatique de ces opérations, sans donner trop de détails, car cet aspect sera traité de manière plus approfondie lors de l'exposé de la plate-forme $\mathcal{IKBS}$. L'ensemble des opérations de base qui sont applicables aux connaissances représentées dans le formalisme CoDesc , sont implantées dans le langage orienté-objet Java.

Dans un deuxième temps, nous donnons un éventail des opérations sur les valeurs, sur les concepts et développerons un modèle permettant de gérer l'évolution des objets.

3.2 Eléments de conception

Nous introduisons ici des éléments concernant l'implantation des opérations de base dans un langage de programmation orienté-objets (LPOO). Le modèle CoDesc est en effet le coeur de la

plate-forme $\mathcal{IKBS}$, développée dans le LPOO `Java`, sous la forme d'un ensemble de classes du langage, organisées en une hiérarchie unique. Cette hiérarchie de classes du langage est le **méta-modèle** de CoDesc : la modification de ces classes entraine une modification des fonctionnalités et de la sémantique du modèle. L'architecture globale du système est donc constituée de trois niveaux :

- Un *méta-modèle* implanté sous forme d'une hiérarchie de classes d'un langage orienté-objets (Java).
- Les instances de ces classes (au sens langage) représentent les connaissances modélisées par les experts, en particulier les modèles descriptifs (concepts) et les cas (descriptions).
- Les cas (descriptions) sont instances (au sens représentation) des modèles descriptifs.

Remarque : le lien « est-un » qui lie une description à un concept, n'est pas implanté par le lien d'instanciation qui lie un objet à sa classe dans le langage objet. La raison est double, d'une part les concepts ne sont pas implantés à l'aide de classes du langage, afin de pouvoir être définis et manipulé au niveau applicatif par l'expert ; et d'autre part, car le rôle du lien d'instanciation n'est pas exactement le même en programmation qu'en représentation : le lien qui relie un objet à une classe ne peut être changé dans un langage objet, alors qu'il évolue fréquemment en représentation des connaissances. La classification d'instances dans les SRCOs vise justement à positionnner une instance dans une hiérarchie de classes préexistante (§ 1.11.2).

Autre remarque : De la même manière, le lien de spécialisation (« sorte-de ») qui lie deux concepts ne correspond pas au lien d'héritage utilisé dans les LPOOs.

3.2.1 Choix de conception

La représentation de valeurs complexes telles que nous les avons abordées au chapitre précédent nécessite de définir des nouveaux types de données informatique adapté pour les représenter. En effet, les types de base prédéfinis dans les langages de programmation sont généralement insuffisants. Un exemple simple peut-être donné par le type *Entier* (Integer), natif dans tout langage de programmation, qui permet de représenter des valeurs entières simples mais pas des intervalles d'entiers par exemple. Il est donc nécessaire de définir un type de données *intervalle* et les opérations arithmétiques permettant de manipuler ce type de données.

Parallèlement, la mise en oeuvre d'opérations formelles sur des valeurs complexes nécessitent un type de données appropriée pour être efficace. Il est cependant rare de pouvoir disposer d'une structure de données adaptée à tout type d'opération. Parfois une structure de données sera plus adaptée qu'une autre pour l'insertion d'un nouvel élément dans un arbre ou un graphe, mais pas pour la recherche ou pour la suppression, etc. Il est donc intéressant de rendre indépendante l'implantation des opérations formelles (fonctions de haut niveau) par rapport à la structure de données sous-jacente, de manière d'une part à pouvoir faire évoluer la structure de données sans remettre en cause les opérations formelles et d'autre part pouvoir éventuellement disposer de plusieurs structures de données alternatives, choisies en fonction du contexte, pour leur efficacité dans telle ou telle situation.

Types Abstrait de Données

Une manière élégante d'encapsuler la (ou les) structure de données est de définir un ensemble d'opérations *primitives* dépendantes de cette structure, puis de définir l'ensemble des opérations formelles à l'aide de ces primitives, sans jamais accéder directement à la nature réelle de la structure de données sous-jacente, ce qui revient à définir un Type de Données Abstrait (ou TDA) (Abelson et al. 1989) (cf. fig. 3.1).

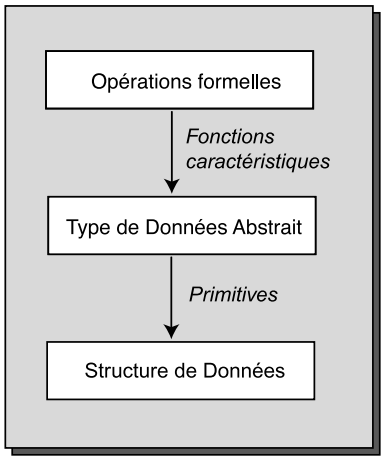


FIG. 3.1 – *Type de Données Abstrait.*

Types de Données Abstrait et classes des langages orientés-objets

La définition de Type Abstrait de Données trouve une implantation naturelle dans les langages de programmation orientés-objets (LPOO) (cf. 1.11). En effet, un des principes fondamentaux de la conception objet de programmes consiste à encapsuler des variables et des fonctions au sein de structures particulières : les classes. La visibilité des variables et des fonctions (appelées méthodes dans ce contexte) peut être définie comme *publique* ou *protégée* par exemple, pour être visibles ou masquées (resp.) au regard des autres classes [61].

Ainsi, par définition des langages objets, les classes sont particulièrement bien adaptées pour modéliser un TDA : la (ou les) structures de données sont des variables de la classe, les fonctions *primitives* sont des méthodes privées, les *fonctions caractéristiques* sont des méthodes publiques (fig. 3.2) qui accèdent à la structure de données par l'intermédiaire des primitives.

Hiérarchies de classes et polymorphisme

Les langages orientés-objets offrent par ailleurs certains mécanismes très puissants, tels que l'organisation hiérarchique (multiples ou non) des classes et la *liaison dynamique* des objets et des méthodes, ce qui autorise la définition de méthodes *polymorphes*, c'est-à-dire de fonctions qui prennent des sens différents en fonction du contexte d'application, défini par la classe de l'objet

61. Plus exactement, au regard des instances des autres classes.

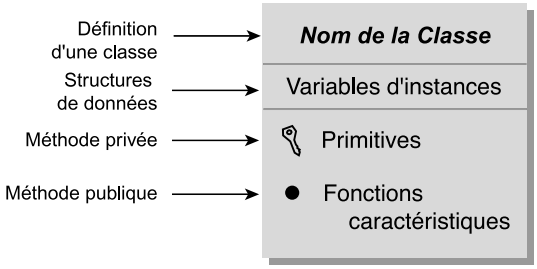


FIG. 3.2 – *Une classe représente un Type de Données Abstrait*

auquel la méthode s'applique[62].

Conception orientée-objets et valeurs complexes

Dans notre contexte, nous souhaitons élaborer un modèle informatique pour représenter différents types de valeurs munis d'opérations formelles, qui soient en particulier :

- **extensibles** : de nouveaux types de valeurs doivent pouvoir être introduits aisément, par définition complète ou par extension de types existants.
- **polymorphes** : des ensembles de valeurs hétérogènes doivent pouvoir être manipulés globalement c'est-à-dire sans nécessairement connaître le type précis de chaque valeur contenue.

L'utilisation de la structuration en une hiérarchie de classe et du polymorphisme est très utile, car ils donnent la possiblité d'effectuer certaines opérations sur les valeurs sans se soucier de leur type précis, qui peut être défini *a posteriori* (condition d'extensibilité). Ces opérations s'appliquent cependant correctement en fonction de la spécificité du type réel, par le biais du mécanisme de *liaison dynamique* des objets et des méthodes. De plus, les opérations communes à différents groupes de types de valeurs peuvent ainsi être *factorisées* au sein de classes abstraites, communes à ces différents groupes.

La figure 3.3 illustre un exemple de hiérarchie de classes pour représenter deux types de valeurs : des entiers et des intervalles. La racine de la hiérarchie est la classe abstraite *Valeur* qui définit l'interface des opérations possibles sur tout type de valeur. La méthode *Egal(Valeur)* par exemple est disponible pour tout type de valeur et s'instanciera selon le contexte d'application : comparaison de deux entiers[63] ou de deux intervalles. Notons que chaque TDA ainsi modélisé par des classes possède une structure de données adaptées à la valeur dénotée : type *int* (integer) natif pour les entier, deux *int* pour les intervalles correspondant aux bornes inférieure et supérieure. Le type *Intervalle* possède ici une primitive *Ordonner()*, qui réordonne la borne inf et la borne sup, si $sup < inf$ (cohérence interne), une fonction caractéristique *BorneInf()*, qui retourne la borne inférieure, etc.

62. à ne pas confondre avec le polymorphisme dit *ad hoc* qui consiste à choisir, parmi un groupe de fonctions de même nom (mais de signatures différentes), celle qui correspond aux paramètres transmis lors de l'appel à la fonction. Ce type de polymorphisme existe dans des langages non-objets, tel que le langage `C` par exemple.

63. le premier étant l'objet-entier auquel la méthode égal s'applique, le second étant la valeur passée en paramètre de la fonction *égal*.

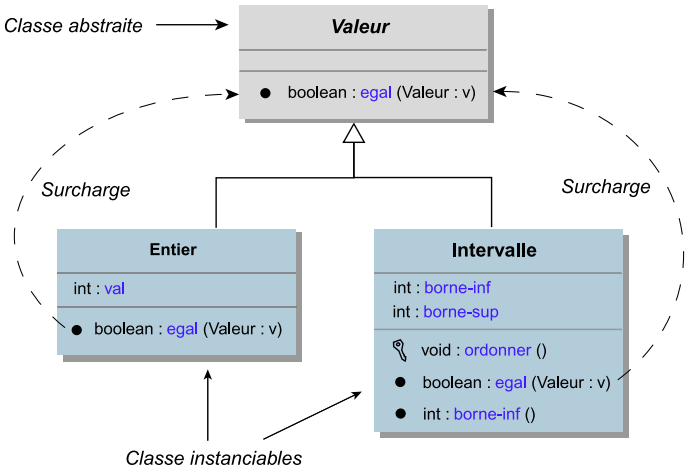


FIG. 3.3 – *Exemple d'une hiérarchie de types de valeurs*

Dans la hiérarchie complète des valeurs (illustrée à l'annexe D), des niveaux intermédiaires sont considérés, tels que valeur *qualitative*, qui regroupe les types nominaux ou hiérarchiques, valeur *quantitative* (discrète ou intervalle), valeur de composant ou encore valeur *inconnu* par exemple.

Conclusion sur la conception

La conception que nous avons retenues pour l'implantation des valeurs complexes du modèle CoDesc peut se résumer dans les points suivants :

- Les différents types de valeurs sont des Types de Données Abstrait représentés par des classes du langages de programmation,
- Les primitives du TDA sont des méthodes privées de la classe, les fonctions caractéristiques sont des méthodes publiques. Ces dernières correspondent aux opérations formelles possibles sur les valeurs,
- Les types de valeurs sont organisées en une hiérarchie de classes, de racine la classe abstraite *Valeur*, qui définit l'interface des opérations possibles sur tout type de valeurs. Des niveaux intermédiaires sont introduits, tels que *Valeur quantitative* (dont les sous classes sont *entier*, *intervalle*, etc.), type *ensemble*, valeur inconnu, etc.

3.3 Opérations sur les valeurs

Nous souhaitons disposer d'un ensemble d'opérations formelles sur les différents types de valeurs complexes, telles que nous les avons exposées au chapitre précédent. Ces opérations permettent par exemple de calculer l'intersection ou l'enveloppe convexe d'intervalles de valeurs, l'union ou la différence de deux ensembles, ou encore comparer des valeurs hiérarchiques, etc.

Le but poursuivi est d'encapsuler la complexité de ces opérations de base, au regard des différentes méthodes de comparaison, d'identification ou de classification, qui sont des opérations de plus haut niveau. Celles-ci sont exposées dans la seconde partie de cet exposé. L'objectif est

ainsi de disposer d'une « boîte à outils » permettant de faciliter la conception et l'implantation de méthodes d'analyse et de mieux maîtriser leur complexité algorithmique.

3.3.1 Opérations ensemblistes

Comme nous l'avons développé au § 2.9, un attribut est défini comme étant une application à valeur dans l'ensemble des parties de son domaine. Il en résulte que tout type de valeur peut être vu comme un ensemble. Une valeur de type simple v peut être interprétée comme le singleton $\{v\}$; la valeur *inconnu*, comme le domaine de définition de l'attribut, ou encore la valeur vide (non renseignée) comme l'ensemble vide.

Deux types d'ensembles sont considérés : ensembles de valeurs quantitatives et ensembles de valeurs qualitatives, certaines opérations étant spécifiques aux valeurs quantitatives, comme le calcul de la moyenne par exemple.

Le tableau 3.1 présente les opérations formelles applicables aux valeurs de type ensemble et donc à tout type de valeur, par l'intermédiaire d'une transformation préalable de mise sous forme ensembliste. Deux types de notations sont utilisées : notation fonctionnelle et notation mathématique.

Fonction	**Notation**	**Commentaire**
Egal(v_1, v_2)	$v_1 = v_2$	Tester l'égalité des valeurs.
Card(v)	$\|v\|$	Retourne le cardinal de l'ensemble. Retourne 1 lorsque v est simple.
Union(v_1, v_2)	$v_1 \cup v_2$	Calcule l'union de v_2 (valeur simple ou ensemble de valeurs) avec v_1.
Intersection(v_1, v_2)	$v_1 \cap v_2$	Calcule de l'intersection des deux valeurs.
Difference(v_1, v_2)	$v_1 - v_2$	Différence de v_1 avec v_2, éliminer les éléments de v_1 présents dans v_2.
DiffSym(v_1, v_2)	$v_1 \Delta v_2$	Différence symétrique de v_1 avec v_2, équivalent formellement à $(v_1 \cup v_2)$ - $(v_1 \cap v_2)$.
Appartient(v_1, v_2)	$v_2 \in v_1$ ou $v_2 \subseteq v_1$	test d'appartenance (v_2 est simple) ou d'inclusion (v_2 est un ensemble).
PlusGeneral(v_1, v_2)	$v_2 \leqslant v_1$	Pour tester si v_1 est une valeur plus générale que v_2 (dans le treillis de valeurs), c'est-à-dire si v_1 recouvre v_2. Par exemple, $v_1 = \{[1\ 5], [10\ 15]\}$ recouvre $v_2 = [12\ 13]$. Ne pas confondre avec la relation $\leq$ traduisant l'ordre usuel sur les nombres ou les valeurs ordinales.

TAB. 3.1 – *Les opérations formelles sur les ensembles (fonctions caractéristiques).*

Du point de vue du modèle informatique, les ensembles de valeurs sont représentés par l'intermédiaire d'un Type de Données Abstrait (TDA) **ensemble**, appelé *SetValue*, qui est une spécialisation du type *Value*, le type générique. Le tableau 3.2 présente quelques primitives du TDA ensemble sur lesquels sont basées les opérations formelles.

Notons que dans une conception orientée-objet, toutes les opérations sont des *méthodes* des classes qui représentent les valeurs (Value, SetValue, etc.). Le premier paramètre de chaque fonction présentée est l'instance à laquelle s'applique la méthode.

Méthodes	**Commentaires**
asSet(v)	Transforme v en une valeur de type ensemble.
add(v_1,v_2)	ajout des éléments de v_2 dans v_1. v_2 est ajoutée à v_1 s'il est une valeur simple qui n'est pas déjà présente dans v_1 (égalité stricte). Si v_2 est un ensemble, les valeurs simples contenues sont ajoutées une à une. Correspond à l'opération d'union du type de données ensemble usuel.
remove(v_1,v_2)	Si v_2 est simple, éliminer v_2 de e. Si v_2 est un ensemble, éliminer de v_1 toute occurrence des valeurs contenues dans v_2.
Generaliser(v_1, v_2)	Calcule la valeur généralisée la plus spécifique des deux valeurs v_1 et v_2. Par exemple, si v_1 = {5, [10 15]} et v_2 = [12 17], alors la valeur généralisée est {5, [10 17]}.
Normaliser(v)	Normaliser v consiste d'une part à réordonner les valeurs si l'attribut est munie d'une relation d'ordre total, d'autre part à éliminer les valeurs généralisées par des valeurs présentes dans v. Si v = {[3 5], 1, 4}, *Normaliser(v)* donne {1, [3 5]}

TAB. 3.2 – *Quelques primitives des valeurs de type* ***Ensemble****.*

3.3.2 Opérations sur les valeurs quantitatives

Certaines opérations sont spécifiques aux valeurs quantitatives. Dans la même logique de conception que pour les valeurs de type ensemble, toute valeur quantitative peut être interprétée comme un intervalle. Une valeur discrète v peut être interprétée comme l'intervalle $[vv]$.

Le tableau 3.3 présente quelques opérations usuelles sur les ensembles de valeurs quantitatives.

Méthodes	**Commentaires**
bInf(v)	Retourne la borne inférieure de v. Retourne v si elle est simple, la borne inférieure si v est un intervalle et le plus petit élément, si v est un ensemble.
bSup(v)	Retourne la borne supérieure de v
average(v)	Retourne la moyenne des valeurs contenues dans v.
extremity(v)	Retourne l'ensemble formé par les extrémités de v. Par exemple, si v = [5 10], l'ensemble formé est {5, 10}.
disjointed(v_1,v_2)	Teste si v_1 et v_2 sont disjointes (intersection vide).

TAB. 3.3 – *Opérations sur les ensembles de valeurs quantitatives.*

Notons que les opérations arithmétiques de base, addition, soustraction, etc. sont définis au niveau des valeurs quantitatives simples.

3.3.3 Conclusion sur les opérations concernant les valeurs

Nous avons esquissé un ensemble d'opérations de base pour manipuler des ensembles de valeurs quantitatives et qualitatives, ainsi qu'un aperçu du modèle informatique réalisé pour l'implantation du modèle CoDesc et des opérations associés aux valeurs complexes.

Ces opérations interviennent dans de nombreux algorithmes, tels que la recherche des APS, la synthèse de descriptions généralisantes, ou encore dans la méthode d'identification développée au chapitre 4.

Dans la suite de ce chapitre, nous détaillons quelques unes de ces opérations internes au modèle.

3.4 Recherche de l'Ascendant le Plus Spécifique (APS)

Trouver un concept dans une hiérarchie simple existante qui caractérise au mieux un ensemble d'individus est très utile dans de nombreuses méthodes de traitement des connaissances. Cela permet de déduire les propriétés partagées par tous et donc d'établir une base commune pour comparer, identifier ou classer ces individus. Ce concept est appelé l'Ascendant le plus spécifique, car il généralise tous les individus. Il n'existe par ailleurs aucun autre concept défini plus précis qui généralise tous les individus.

Le problème n'est pas si simple lorsque le nombre de concepts et d'individus mis en jeu est important. Il se complique d'avantage Lorsque la hiérarchie de concept est multiple, c'est-à-dire lorsque les individus sont attachés à plusieurs catégories. Il existe alors plusieurs APS. Dans le modèle proposé (cf. chap.2), nous ne considérons que des hiérarchies simples. L'algorithme que nous donnons est cependant applicable au cas général de hiérarchies multiples.

3.4.1 Contexte et définition

La recherche des Ascendants les Plus Spécifiques (APS) dans une *hiérarchie multiple* [64] est une opération classique dans les modèles de représentation par objets et les logiques terminologiques (Baader et al. 1994, Napoli 1994). Elle est à la base du mécanisme de classification qui consiste à positionner un objet X, classe ou instance, dans une hiérarchie de classes. C'est la première étape d'un mécanisme plus complexe qui consiste à :

1. rechercher les Ascendants les Plus Spécifiques de X,
2. recherche des Descendants les Plus Généraux (DPG de X), lorsque X est une classe,
3. puis enfin, mise en place des nouvelles relations entre X, ses ascendants et ses descendants.

Le problème de la recherche du (ou des) Ascendants les Plus Spécifiques, noté APS, dans une hiérarchie simple (ou multiple) peut être étendu à des ensembles. Dans la définition suivante, X peut être indifféremment un objet, un ensemble d'objet et/ou de concepts (classes) de la hiérarchie considérée :

Définition 3.1 ***Recherche des APS***

64. Dans une **hiérarchie multiple**, un individu ou un concept peut possèder plusieurs ascendants.

Étant donné :

- *un ensemble X, composé d'individus de Ω ou de concepts de $\mathcal{H}$,*
- *une hiérarchie $\mathcal{H}$ contenant un ensemble de concept $\mathcal{C}$,*
- *un prédicat permettant d'établir l'appariement entre un élément de X et un concept de $\mathcal{H}$.*

Rechercher : *Le concept le plus spécifique qui généralise tout élément de X.*

3.4.2 l'APS dans CoDesc

Dans notre modèle, les individus appartiennent à une hiérarchie de concepts de type simple, c'est-à-dire que chaque concept possède un ascendant unique. L'APS est donc unique et son existance est garantie par le fait que H est munie d'un élément maximale $\mathcal{C}_0$. La recherche de l'APS consiste alors à trouver, à partir d'un ensemble hétérogène d'individus [65], le concept existant de $\mathcal{H}$ le plus spécifique, dont l'extension recouvre tous les individus considérés.

Le prédicat de généralisation

Nous nous plaçons dans un cadre général de recherche de l'APS, l'ensemble X considéré étant hétérogène (individu ou concept). Dans ce contexte, il est nécessaire de disposer d'un *prédicat générique* fonctionnant indifféremment sur les individus et les concepts [66]. Ce prédicat est utile dans trois situations :

- Déterminer si un individu est plus général qu'un autre individu. En particulier, pour un concept donné, il existe un individu maximal qui généralise toute autre description du concept. La description de cet individu est telle que tout descripteur est renseigné par la valeur *inconnu*. En effet, pour tout descripteur $A \in \mathcal{A}$, la valeur *inconnu* est la plus générale (l'élément maximal) dans le treillis de valeurs associé au co-domaine de A (cf. § 2.9). Par extension, un individu ω_1 généralise un autre individu ω_2, noté $\omega_2 \preccurlyeq \omega_1$ si et seulement si : $\forall A \in \mathcal{A}$, $A(\omega_2) \leqslant A(\omega_1)$ (au sens $\omega_2 \subseteq \omega_1$). Ainsi, l'opérateur de généralisation sur les individus d'un concept $\mathcal{C}$ est définit comme le produit des opérateurs de généralisation sur l'ensemble des attributs $A \in \mathcal{A}$. Cette relation n'est pas exploitée pour la recherche de l'APS (ou des APS), mais intervient pour la synthèse de descriptions généralisantes.
- Déterminer si un concept $\mathcal{C} \in \mathcal{H}$ recouvre la description d'un individu $\omega \in X$. Ceci est équivalent au **test d'instanciation** des modèles objets, qui détermine si un individu est instance d'un concept considéré. Notons que dans l'affirmative, il est également instance de tout concept subsumant $\mathcal{C}$. L'individu et le concept sont alors liés par une relation d'instanciation (lien est-un). Par extension, nous notons $\omega \preccurlyeq \mathcal{C}$ pour signifier que ω est instance de $\mathcal{C}$.
- Déterminer si un concept est plus général qu'un autre, ce que l'on note $\mathcal{C}_1 \preccurlyeq \mathcal{C}_2$, qui exprime que $\mathcal{C}_2$ est plus général que $\mathcal{C}_1$.

65. **Ensemble hétérogène d'individus** : qui appartiennent à des concepts différents.
66. Le prédicat de généralisation est parfois appelé `covers()` dans les systèmes qui manipulent des concepts.

Ainsi, nous considérons que l'opérateur binaire de généralisation, noté $\preccurlyeq$, qui traduit la relation de généralisation (et par extension, également la relation d'instanciation qui lie un individu à un concept), est générique dans le sens où il permet indifféremment de comparer des individus et des concepts.

Nous pouvons étendre encore davantage cet opérateur, en considérant des ensembles d'objets.

Définition 3.2 ***L'opérateur de généralisation étendu à des ensembles***

Soit $X = \{x_1, \ldots, x_i, \ldots, x_n\}$, un ensemble d'individus de Ω et/ou de concepts de $\mathcal{H}$, et $\mathcal{C}$ un concept de $\mathcal{H}$. Nous définissons l'opérateur $\preccurlyeq_G$ par extension à des ensembles de l'opérateur $\preccurlyeq$, tel que :

$$X \preccurlyeq_G \mathcal{C} \Leftrightarrow \forall x \in X,\ x \preccurlyeq \mathcal{C}$$

Propriétés de l'APS

L'Ascendant le Plus Spécifique généralise tout élément de X. La définition précédente de l'opérateur de généralisation étendu à des ensembles hétérogènes d'individus et de concepts conduit à exprimer simplement cette condition de la façon suivante :

Proposition 3.1 *L'APS généralise tout élément de X :*

$$X \preccurlyeq_G APS \Leftrightarrow \forall x \in X,\ x \preccurlyeq APS$$

Ce qui est une *condition nécessaire* mais non suffisante. En effet, rien ne garantit dans cette proposition que l'APS est le plus spécifique, un certain nombre de concepts peuvent vérifier cette condition : tout concept qui généralise X. Soit L_A, l'ensemble des concepts qui généralise X. La proposition suivante complète la définition :

Proposition 3.2 *Etant donnée L_A la liste de tous les concepts qui généralisent X, l'APS recherché est un concept $APS \in L_A$, tel que :*

$$\forall \mathcal{C} \in L_A,\ APS \preccurlyeq \mathcal{C}$$

L'unicité de la solution est garantie par le fait que la hiérarchie $\mathcal{H}$ considérée est simple. En effet, dans ce cas, tout concept possède un unique ascendant. Par contre, dans une hiérarchie multiple de classes ou de concepts, l'APS n'est pas nécessairement unique.

Les algorithmes suivants donnent le moyen de trouver l'APS recherché lorsque la hiérarchie $\mathcal{H}$ est simple ou multiple.

3.4.3 Algorithme de recherche des APS, par une approche récursive descendante

Dans une hiérarchie simple $\mathcal{H}$, l'algorithme 1 recherche l'Ascendant le plus Spécifique d'un ensemble hétérogène X, constitué indifféremment d'individus $\omega \in \Omega$ et/ou de concepts de $\mathcal{C} \in \mathcal{H}$, en se basant sur la structure de la hiérarchie pour effectuer la recherche.

Principe

L'algorithme 1 effectue un parcours en profondeur de la hiérarchie $\mathcal{H}$, récursivement à partir d'un concept $\mathcal{C}$. $\mathcal{C}$ peut-être initialisé à $\mathcal{C}_0$, ou bien être la racine d'une hiérarchie locale $\mathcal{H}'$.

A chaque étape, le test de généralisation est effectué pour déterminer si $\mathcal{C}$ est un ascendant de X. Si ce n'est pas le cas, $\emptyset$ est retourné. Sinon, la fonction est appelée récursivement sur chaque descendant direct de $\mathcal{C}$, afin de déterminer si $\mathcal{C}$ est le plus spécifique.

L'algorithme se fonde sur les deux fonctions suivantes :

- DESC($\mathcal{C}$) retourne la liste des descendants directs de $\mathcal{C}$, les fils au sens de la spécialisation.
- nous considérons ici le test $X \preccurlyeq_G \mathcal{C}$, qui est vrai si $\mathcal{C}$ généralise tout élément de X.

Algorithme 1 Rechercher-APS($\mathcal{C}$, X), Rechercher l'APS par une approche descendante

Données : Un concept $\mathcal{C}$, l'élément maximal d'une hiérarchie locale $\mathcal{H}' \subseteq \mathcal{H}$. Un ensemble X d'individus ou de concepts dont on recherche l'APS.

Résultat : L'ensemble vide $\emptyset$ si aucun APS de X n'existe dans $\mathcal{H}'$. La liste L_{APS} contenant l'APS recherché (ou les APS dans une hiérarchie multiple) de $\mathcal{H}'$ généralisant X sinon.

$L_{APS} \leftarrow \emptyset$
si $X \preccurlyeq_G \mathcal{C}$ **alors**
  pour tout $\mathcal{C}_i \in DESC(\mathcal{C})$, descendants directs de $\mathcal{C}$ **faire**
    $L_{APS} \leftarrow L_{APS} \bigcup$ Rechercher-APS($\mathcal{C}_i$, X)
  fin pour
  si $L_{APS} = \emptyset$ **alors**
    $L_{APS} \leftarrow \mathcal{C}$ [$\mathcal{C}$ est un APS]
  fin si
  retourner L_{APS}
sinon
  retourner $\emptyset$
fin si

Intérêts et limites

Cet algorithme nécessite une hypothèse initiale : le choix de $\mathcal{C}'$, qui caractérise le sommet de la hiérarchie locale $\mathcal{H}'$. En effet, il est coûteux de parcourir l'ensemble des concepts lorsque $\mathcal{H}$ est de taille importante : le choix initial de $\mathcal{C}$ a donc pour conséquence de diminuer l'espace des concepts à prendre en compte. Cette hypothèse est donc également une *heuristique*.

Par ailleurs, d'un point de vue conception, cet algorithme est bien adapté à une implantation dans un langage orienté-objet. En effet, on peut considérer que le concept maximal de la hiérarchie locale $\mathcal{H}'$ est au centre de la recherche de l'APS. C'est lui qui guide le processus et retourne l'ensemble vide lorsqu'il n'est pas lui-même un ascendant de X, et retourne l'APS trouvé sinon. Il est donc naturel, dans une approche objet, qui est celle que nous avons adoptée pour la

conception de la plate-forme $\mathcal{IKBS}$, développée dans le langage de programmation orienté-objet `Java` (LPOO), d'implémenter l'algorithme 1 comme une *méthode* [67] associée aux concepts. Plus précisément, dans notre implantation, les concepts sont les instances (au sens du LPOO) d'une classe appelée *Schéma*. Ainsi, chaque concept (représenté par une instance de la classe *Schéma*) est à même de trouver l'APS à partir d'un ensemble hétérogène constitué d'individus ou de concept et l'opération devient par la même une fonctionalité intrinsèque au modèle CoDesc. Nous reviendrons sur la conception orienté-objet de la plate-forme $\mathcal{IKBS}$ au chapitre 6.

D'autre part, l'algorithme peut également s'appliquer à une hiérarchie multiple de classes ou de concepts. Dans ce cas, la liste L_{APS} construite possède plusieurs concepts, l'APS n'est pas nécessairement unique. Une heuristique supplémentaire peut alors être utilisée, qui consiste à marquer chaque concept visité par $Vrai$ s'il est un APS potentiel et par $Faux$ s'il ne généralise pas X. En effet, dans l'algorithme 1 chaque élément qui possède plusieurs ascendants directs est visité plusieurs fois, ce qui est inutile. Simon (1995) et Simon (2000) proposent un algorithme de recherche des APS utilisant un tel marquage. Il suffit simplement d'introduire le marquage dans l'algorithme 1 aux vues du résultat du test $X \preccurlyeq_G \mathcal{C}$.

Il peut également être intéressant, au préalable de l'activation de l'algorithme, d'effectuer un pré-traitement sur X, dans le but de détecter si pour un individu de X, le concept le plus spécifique pour cet élément est déjà connu, puis substituer l'individu par ce concept dans X. L'intérêt de ce pré-traitement est double :

- Diminuer éventuellement la cardinalité de X. En effet, si plusieurs individus sont attachés au même concept, ils seront substitués par ce concept. Diminuer le nombre d'élément de X présente un intérêt évident en terme d'efficacité algorithmique.
- Si le concept est connu à l'avance pour chaque individu, X est alors uniquement consitué de concepts de $\mathcal{H}$.

Lorsque le second point est vérifié, nous privilégions une approche *ascendante* qui permet d'éviter le choix initial du concept $\mathcal{C}$ dans $\mathcal{H}$.

3.4.4 Algorithme de recherche des APS d'un ensemble de concepts par une approche ascendante

Tout élément de X est un concept de $\mathcal{H}$, simple ou multiple. L'algorithme proposé recherche les APS de manière ascendante, à partir des concepts de X.

Principe

L'algorithme 2 fonctionne sur le type de données abstrait liste (au sens du langage fonctionnel $LISP$). La liste L est initialisée à X. Le principe consiste à substituer chaque élément de L qui n'est pas ascendant de tous les autres (trop spécifique) par son (ou ses) ascendants. On réitère le processus, récursivement, jusqu'à ce qu'il ne reste que des ascendants, qui sont également des APS.

67. Les **méthodes** sont les fonctions encapsulées aux sein des classes du langage de programmation.

L'algorithme utilise les fonctions suivantes :

- ASC($\mathcal{C}$) retourne la liste des ascendants directs de $\mathcal{C}$ dans une hiérarchie $\mathcal{H}$, simple ou multiple.
- CAR(L) retourne le premier élément de la liste.
- CDR(L) retourne la liste privée du premier élément.
- l'opérateur d'union ensembliste $\bigcup$ permettant de fusionner deux listes en ne gardant qu'une seule occurrence de chaque élément.

Algorithme 2 RechercherAPS-3(L) : rechercher les APS de X par une méthode ascendante

Données : Un ensemble X de concepts et une hiérarchie $\mathcal{H}$. La recherche est effectuée par l'appel initial RechercherAPS-3(X)

Résultat : L'ensemble vide $\emptyset$ si aucun APS de X n'existe dans $\mathcal{H}$. La liste L contenant l'APS recherché (ou les APS dans une hiérarchie multiple) de $\mathcal{H}$ généralisant X sinon.

si $L = \emptyset$ **alors**
  retourner L
sinon
  si CDR(L) $\preccurlyeq_G$ CAR(L) **alors**
    $L \leftarrow$ CAR(L) $\bigcup$ RechercherAPS-3($CDR(L)$) [CAR(L) est un APS]
    retourner L
  sinon
    $L \leftarrow$ ASC(CAR(L)) $\bigcup$ CDR(L) [CAR(L) est remplacé par la liste de ses ascendants directs]
    retourner RechercherAPS-3(L) [Le processus est réitéré]
  fin si
fin si

Discussion

Cette stratégie correspond à un *raisonnement inductif*, qui part des éléments de plus bas niveau pour rechercher les concepts suffisamment génériques. D'autre part, il ne nécessite pas d'hypothèse sur le concept initial, qui peut conduire à un échec comme nous l'avons souligné. Enfin, une seule variable de type liste est utilisée, la structure de la hiérarchie étant exploitée par la fonction ASC($\mathcal{C}$).

Comparativement à l'approche précédentes, celle-ci requiert une connaissance préalable sur les individus considérés, celle de connaître les concepts les plus spécifiques pour chaque individu. Ce qui est le cas lorsqu'une base d'individus de référence (pré-classifiés) est utilisée. En effet, dans notre modèle, toute description est associée à un concept spécifique, même si sa classe précise (valeur de l'attribut de classe) n'est pas nécessairement connue. Dans ce cas, la stratégie 2 est utilisée, par défaut, c'est la stratégie 1.

3.4.5 Intérêt en Systématique

Pour notre problématique, trouver l'APS présente un intérêt en particulier pour l'identification, dès lors que l'utilisateur n'est pas en mesure de définir *a priori* le concept auquel est attaché l'individu à identifier. En effet, le problème de l'identification se pose d'une manière quelque peu différente, selon que l'utilisateur est expert du domaine ou néophyte.

L'expert peut généralement déterminer rapidement le concept qui caractérise un taxon de haut niveau (la famille ou l'espèce) auquel appartient l'individu à identifier. Sa connaissance du domaine lui permet, par un raisonnement synthétique global, d'identifier immédiatement la famille (et donc un concept). Dans ce cas, la méthode d'identification a pour pour objectif de préciser le taxon du spécimen étudié jusqu'au niveau de l'espèce[68], sur la base d'un ensemble *homogène* d'individus, c'est-à-dire qui appartiennent au même concept. La recherche de l'APS n'est alors pas utile, puisque toute les descriptions sont toutes comparables, elles ont la même structure, et les mêmes descripteurs. Dans le second cas, soit l'expert définit un ensemble de concept et de descriptions appartenant à ces concepts, soit aucune hypothèse n'est effectuée préalablement. La donnée de l'APS est dans ce cas importante, puisqu'elle permet de comparer un ensemble *hétérogène* d'individus X sur une base commune : le concept le plus spécifique qui généralise tout élément de l'ensemble.

3.5 Opérations de généralisation

La production d'une description de concept à partir d'un ensemble de descriptions individuelles est une forme de généralisation. La généralisation est couramment utilisée par les méthodes d'apprentissage symbolique, surtout en apprentissage supervisé. En particulier, l'apprentissage de concepts est une forme d'apprentissage dont la généralisation d'un ensemble d'individus en un concept est l'opération centrale. Celui-ci a connu un succès considérable ces dernières années, ce qui explique l'abondance de la littérature sur les sujets concernant la généralisation (voir par exemple Mitchell (1997)). Dans tous les cas, la généralisation est une opération *inductive* qui permet de passer du plus spécifique au plus général. Contrairement à la recherche des APS, qui recherche des concepts existants, l'opération de généralisation construit la description d'un nouveau concept dont l'extension recouvre l'ensemble des descriptions individuelles initiales.

3.5.1 Définition

Une définition générale de la généralisation est donnée par Michalski (1983) :

« *La généralisation est une opération qui permet de passer des expressions représentant les observations vers l'expression d'une abstraction de ces observations.* »

Le cadre général d'un problème de généralisation est décrit par Mitchell (1990*a*) comme suit (cf. Valtchev (1999)) :

Définition 3.3 ***La généralisation***

68. ce qui revient à affecter une valeur à l'attribut de classe *Taxon*

Étant donné :

- *un ensemble d'individus* Ω*, décrits à l'aide d'un langage de description des individus,* L_Ω*,*
- *un ensemble de généralisations potentielles, encore appelées hypothèses,* Γ*, les généralisations sont décrites en termes d'un langage de description des généralisations,* L_Γ*,*
- *un prédicat,* `covers()`*, permettant d'établir l'appariement entre une description d'individu et une généralisation.*

Rechercher : *Une généralisation qui s'apparie avec tous les individus.*

D'autre part, comme le souligne Mitchell (1990*b*), il existe plusieurs généralisations possibles, donc plusieurs concepts, pour un ensemble d'individus. Il est ainsi possible de définir une relation de généralité $\mathcal{R}$ sur un couple de concepts ($\mathcal{C}_1$,$\mathcal{C}_2$) généralisés à partir du même ensemble d'individus Ω. $\mathcal{C}_1\mathcal{R}\mathcal{C}_2$ traduit le fait que tout individu $\omega \in \Omega$ qui s'apparie avec $\mathcal{C}_2$ s'apparie aussi avec $\mathcal{C}_1$ ($\mathcal{C}_1$ est plus général que $\mathcal{C}_2$).

Dans (Valtchev 1999) la relation de généralité est considérée comme une relation d'ordre en prenant comme équivalentes toutes les généralisations couvrant le même ensemble d'individus. Ainsi, les membres de Γ sont organisés par la relation $\mathcal{R}$ en une structure hiérarchique appelée *espace de généralisations.*

Le sens donné donné à $\mathcal{R}$ dans ce contexte rejoint celui de la généralisation des concepts au sens de CoDesc. L'expert modélise un domaine en choisissant d'expliciter certains concepts parmi un ensemble de représentations possibles, selon des critères d'adéquation avec la réalité. Les concepts sont ensuite organisés en une hiérarchie $\mathcal{H}$ et munie de la relation d'ordre $\preccurlyeq$, telle que $\mathcal{C}_1 \preccurlyeq \mathcal{C}_2$ si $\mathcal{C}_2$ est plus général que $\mathcal{C}_1$.

Par opposition à la construction manuelle d'une hiérarchie de concepts $\mathcal{H}$ par un ou plusieurs experts, l'opération de généralisation en apprentissage automatique supervisé peut être vue comme une recherche dans l'espace des généralisations qui optimise un certain critère (Mitchell 1990*a*). En effet, la conséquence directe de la multiplicité des descriptions généralisantes possible est le besoin d'en choisir une. Ce choix est généralement guidé par l'objectif de l'opération : caractériser une seule ou plusieurs classes (généralisation caractéristique), ou bien discriminer une ou plusieurs classes par rapport aux autres (généralisation discriminante).

3.5.2 Contexte

L'opération de généralisation est considérée dans de nombreux formalismes (Valtchev 1999), qu'ils soient structurels, relationnels ou objets.

Formalismes structurels

Pour les formalismes structurels, les individus ont des descriptions qui rassemblent la totalité des connaissances qui les concernent. Chaque description peut être vue comme un *tuple* [69], composé par les valeurs des différents attributs qui constituent la description. Le but de l'inférence de descriptions est alors de déterminer les plages de variation pour chaque attribut partagé par les

69. les **tuples** correspondent aux *enregistrements* des bases de données.

objets à généraliser. Ainsi, dans les systèmes `COBWEB`, `AUTOCLASS` ou encore `WITT`, la généralisation consiste en une *intersection* d'ensembles d'atomes propositionnels [70] décrivant les individus. Une forme plus sophistiquée permet de rendre les descriptions plus compactes en autorisant des disjonctions entre valeurs dans les atomes. Celles-ci expriment la variabilité de la propriété sous-jacente dans les individus caractérisés.

Formalismes relationnels

Dans les langages de représentation relationnel, tels que les langages de prédicats du premier ordre ou les graphes conceptuels, les descriptions sont des noms liés par des relations. La généralisation cherche à isoler la structure commune entre descriptions. La détection d'une telle structure consiste à rechercher le sous-graphe isomorphe, ce qui est un problème NP-difficile.

En logique des prédicats, généraliser consiste à trouver un appariement entre les clauses, ce qui est un problème proche du problème d'isomorphisme entre graphes, donc NP-complet. Soulignons qu'un des objectif de la communauté *Programmation Logique Inductive* (ILP) (Lavrac & S.Dzeroski 1994, Bergadano & Gunetti 1995) consiste à identifier des classes de langages qui admettent des opérations de généralisation efficaces, sur la base des travaux de Plotkin (1970) et de Shapiro (1981). Les systèmes `GOLEM` (Muggleton 1992) ou `FOIL` (Quinlan & Cameron-Jones 1993) par exemple utilisent des mécanismes tels que le généralisant le plus spécifique et la résolution inverse.

Pour les graphes conceptuels, la généralisation d'un ensemble de graphes représentant des individus s'apparente à la recherche du sous-graphe commun maximal, ce qui est également un problème NP-difficile (Garey & Johnson 1979).

Objets

L'objectif est l'inférence d'une description de classe à partir des descriptions de ses instances. Le principe consiste à effectuer une restriction sur chacun des attributs. Le langage de description des individus (L_Ω) est composé par la liste des couples attribut-valeur. L'ensemble des attributs est défini par la classe commune des objets. Le langage de description des généralisations (L_Γ) est composé de listes attribut-type, chaque type étant un sous-type du type initial de l'attribut correspondant dans la classe commune des objets.

On pourra se référer à Valtchev (1999) pour une description plus précise du mécanisme de généralisation dans les objets.

3.5.3 La généralisation dans CoDesc

Notre approche s'apparent à celle adoptée en apprentissage non-supervisé, pour lequel la *généralisation la plus spécifique* est recherchée, c'est-à-dire celle qui fournit la caractérisation la plus précise de l'ensemble des individus. En effet, nous définissons la généralisation comme une opération de synthèse qui extrait l'ensemble des caractéristiques communes d'un groupe d'individus, dans le but de construire une nouvelle description. Cette description peut-être exprimée sous la

70. les atomes propositionnels, de type attribut-relation-valeur, sont appelés **sélecteurs** (Michalski 1983)

forme de la description d'un *individu synthétique* ou sous la forme d'un nouveau concept (dont la description est donnée par un modèle descriptif). L'extension de ce concept couvrant alors l'ensemble des individus. L'opération de généralisation peut également consister à construire un nouveau concept à partir d'un ensemble de concepts, étant donné le sens donné à la relation de spécialisation et la relation inverse, la relation de généralisation, dans notre modèle.

La généralisation dans CoDesc peut donc revêtir plusieurs formes, selon que l'on cherche à caractériser un groupe d'individus ou à construire un concept plus général à partir d'un ensemble de sous-concepts :

1. construire la description d'un *individu synthétique*, à partir d'un ensemble d'individus appartenant à un ou plusieurs concepts. La généralisation est alors exprimée dans le formalisme *objet-attribut-valeur* exposé précédemment pour la représentation des cas mentionné dans la définition 3.3). Ce formalisme correspond au langage L_Ω, le langage d'expression des descriptions. Ce langage est en effet suffisamment riche pour exprimer la variabilité d'un ensemble d'observations, notamment grâce à la possibilité d'exprimer des ensembles de valeurs, des intervalles et des conjonctions.
2. construire la description d'un *nouveau concept*, à partir d'un ensemble d'individus appartenant à un ou plusieurs concepts. La généralisation est alors exprimée dans le langage de représentation des concepts de CoDesc (L_Γ, cf. définition 3.3),
3. construire la description d'un *concept généralisant*, à partir d'un ensemble de concepts existants. Cette opération est très utile pour mettre en place la hiérarchie des concepts par une conception ascendante.

Dans les trois situations, chaque individu est apparié à un concept pré-existant. Lorsque les individus appartiennent à un unique concept, la généralisation (au sens de 2) aura alors pour objectif de caractériser un concept plus spécifique que le concept commun initial. Inversement, lorsque les individus appartiennent à plusieurs concepts, le concept formé (2) ne généralise pas nécessairement tous les concepts, bien qu'il généralise l'ensemble des individus. En effet, la description du concept généralisé dépend de la représentativité des individus et de l'échantillonnage effectué. Le but est alors de trouver une structure alternative, un autre concept que ceux explicités au préalable par l'expert, qui caractérise de manière plus fine l'ensemble des individus, que leur APS par exemple. Enfin, (3) permet de construire un nouveau concept généralisant un groupe de concepts, ce qui est très utile pour organiser les concepts en une hiérarchie par une conception ascendante (approche inductive), selon les contraintes imposées à la relation de spécialisation et de généralisation (cf.2.6.3).

3.5.4 Synthèse de description généralisante

Etant donné un ensemble $\{\omega_1, \ldots, \omega_n\} \in \Omega$ d'individus pré-classifiés ou non, il est relativement aisé de calculer une description synthétique, à partir de la fonction de généralisation sur les valeurs (cf.§ 3.3). La description obtenue est très intéressante, car elle permet de saisir la *variabilité* de chaque descripteur, pour l'ensemble des individus considérés.

Si l'ensemble d'individu est homogène, c'est-à-dire que tout individu appartient à la même classe, la description généralisante offre une vue de la variabilité de la classe en question. Bien

sûr, la qualité de cette caractérisation dépend fortement de la représentativité de l'ensemble initial d'individus.

Exemple

La figure 3.4 illustre une partie de la description généralisante obtenue par synthèse d'un ensemble de 8 descriptions appartenant à l'espèce *Pocillopora damicornis*. Le symbole $*$ indique que toute valeur du domaine de définition de l'attribut est présente dans au moins une description. La valeur $*$ est formellement équivalente à la valeur ? et donc à $\mathbf{dom}(A) \in \mathcal{A}$. Nous avons introduit ce nouveau symbole $*$ qui caractérise la variabilité du groupe d'individus pour l'attribut considéré, afin de permettre à l'utilisateur de distinguer de ?, qui caratérise la présence d'une valeur *inconnu* pour au moins un individu du groupe.

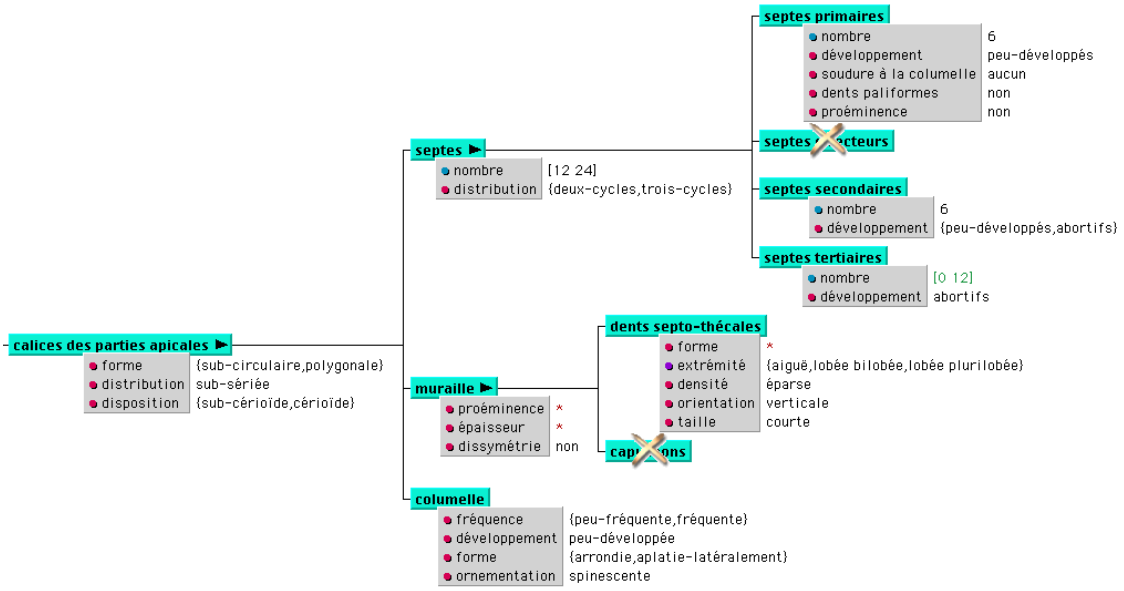


FIG. 3.4 – *Exemple de synthèse de description généralisante obtenue avec $\mathcal{IKBS}$.*

3.6 Gestion de l'évolution des objets

Nous nous plaçons dans cette partie du point de vue du langage orienté-objet Java, support du modèle CoDesc et coeur de la plate-forme $\mathcal{IKBS}$.

Nous présentons un mécanisme générique permettant de gérer l'évolution des connaissances exprimées par l'expert au sein d'une base de connaissances. Toute entité descriptive (concept, composant, descripteurs, etc.) est doté de fonctions liées à la gestion de leur évolution tel que nous l'exposons par la suite.

La nature des objets

Une des caractéristiques fondamentale des langages de programmation à objets (ou orientés-objets) est que l'existence d'un objet est intrinsèquement liée à la classe auquel l'objet appartient. La définition de l'objet est donnée par la classe, au moment de sa création, à l'aide d'une fonction particulière, appelée *constructeur* et un lien structurel (lien *est-un*) qui définit l'appartenance de l'objet à la classe est créé et ne peut jamais être modifié pendant l'exécution du programme.

Dans le même ordre d'idée, pour la plupart des langages [71], lorsqu'une variable est déclarée, un *type de donnée* lui est associé et ne peut être à nouveau défini [72]. La raison principale de cette contrainte provient du fait qu'un espace mémoire est reservé à la déclaration de la variable, calculé sur la base du type de la donnée déclarée. La même variable (le même espace mémoire) est donc inadaptée pour stocker un autre type de variable. La cause est similaire pour les objets, un espace mémoire est réservé, lors de sa création, calculé sur la base de l'occupation mémoire de chaque propriété statique définie au sein la classe de l'objet.

A l'inverse, l'état d'un objet, caractérisé par l'ensemble des valeurs prisent pour chaque attribut de l'objet peut évoluer, dans la mesure ou les modifications de valeurs respectent les types des attributs (on ne peut pas affecter une valeur de type *String* à une variable de type entier). C'est également le cas lorsqu'un attribut définit une relation vers un objet de type C (une classe), la relation doit nécessairement respecter le type de C. L'état des objets évolue donc pendant l'exécution d'un programme et l'état de chaque objet présent en mémoire à un instant donné reflète l'état du système dans sa globalité.

Mais ce qui nous intéresse ici n'est pas la dynamique de changement de l'état des objets, mais celle de l'évolution de sa classe d'appartenance.

Proposition

Changer la nature des objets manipulés par les LPOOs étant impossible, nous proposons de concevoir un mécanisme générique qui « simule » l'évolution des objets. Cette opération est souvent appelé *migration* (Euzenat 1999) dans les systèmes qui manipulent les objets. Par simuler, nous entendons détruire l'objet pour le recréer sous sa nouvelle forme, de façon transparente pour les objets en relation avec lui.

Dans les modèles de représentation des connaissances et en particulier dans les modèles à objets, une instance doit pouvoir évoluer et changer de classe d'appartenance. C'est le cas notament lors d'opérations de la classification d'instance, qui consiste à associer une instance à la classe la plus spécifique (cf. chapitre 3) dans une hiérarchie de classe. Plus généralement, c'est nécessaire lorsque un objet évolue de manière profonde, dans sa structure et qu'il change de type. La responsabilité de gérer les contraintes de cohérence et d'intégrité de ces objets *mutables* est alors à la charge du système.

Le problème de mutation d'objet survient donc dans le cas où l'on souhaite implanter un SRCO à l'aide d'un LPOO, lorsque les concepts du modèle de représentation sont implantés à l'aide des classes du LPOO, ce qui n'est pas nécessairement le cas. L'architecture d'$\mathcal{IKBS}$ en

71. les langages typés

72. Quelques réserves cependant concernant les langages non-typé tel que LISP ou les langages à inférence de type tel que ML ou CAML

particulier distingue plusieurs niveaux de représentation (nous exposerons ces notions au § 6.3) : le modèle CoDesc, implémenté à l'aide de classes du LPOO, les concepts et les cas, représentés par des instances de ces classes. Ce que nous justifions par le fait que les concepts sont modélisés par l'expert à l'aide d'$\mathcal{IKBS}$ et donc pendant son exécution, ce qui ne serait pas possible, si les concepts étaient des classes du langage [73]. Les concepts sont donc bien des objets et non pas des classes, ce qui implique que les liens qui unissent cas et concepts sont gérés au niveau applicatif, indépendamment des contraintes imposées par le langage de programmation.

Cependant, d'autres types de relations internes au modèle CoDesc [74], comme par exemple les relations entre les attributs et leur type qui sont représentés par des classes, sont des relations *is-a* (ou *est-un*), intrinsèques au langage, et le problème se pose alors dans ces termes : comment mettre en place un mécanisme générique permettant de gérer l'évolution du type des objets (la mutation), dans l'objectif de pouvoir modifier dynamiquement les liens entre les objets du langage et le modèle de représentation CoDesc dans essentiellement deux situations :

- au niveau du modèle, lorsqu'un élément descriptif change de type (attribut, composant, etc.). L'expert pourra par exemple changer le type d'un attribut symbolique en type taxonomique, afin d'introduire différents niveaux de précision,
- au niveau des cas, lorsqu'une valeur change de type, mais que l'on souhaite conserver néanmoins un certain nombre de ses caractéristiques (adapter si possible l'ancienne valeur, les liens vers les données contextuelles, les annotations, etc.).

Principe

Notre approche pour « simuler » l'évolution d'objets est basée sur une extension originale de la réplication (clonage) des objets, qui consiste à :

1. Créer une nouvelle instance de même type que l'objet d'origine.
2. Dupliquer les propriétés de l'objet et les liens vers les autres objets (copie superficielle) et éventuellement ces objets eux-même (copie profonde).
3. Déplacer les références qui « pointent » sur l'objet d'origine, vers l'objet répliqué.
4. Détruire éventuellement l'objet d'origine.

Ainsi, une nouvelle instance de même type est substituée à l'objet d'origine. Nous proposons dans un premier temps un patron de conception permettant de gérer de manière transparente la réplication d'un objet a au regard d'un objet (appelé le *client*) effectuant la requête.

3.6.1 Réplication des objets

La réplication d'objets est une opération classique et fréquement nécessaire lorsque l'on souhaite construire un objet à partir d'un autre. On distingue dans cette opération deux alternatives :

- la **copie superficielle**. L'objet est dupliqué, ainsi que les propriétés de type simple et les liens vers d'autres objets. Mais on ne copie pas les objets liés : l'opération n'est pas

73. La classe des langages que nous considérons ici est la classe des langages compilés et non interprétés
74. Ces opérations concernent en fait le méta-modèle CoDesc(§ 6.3).

récursive. Ainsi l'objet d'origine et l'objet répliqué partage des références communes vers les objets liés.

– la **copie profonde**. On duplique l'objet, les propriétés de type simple, les liens vers d'autres objets, ainsi que les objets liés eux-mêmes, récursivement. Une copie profonde est souvent difficile à contrôler, dans le cas de références circulaires ou co-références entre objets. L'opération peut dans ce cas ne jamais terminer et créer une « infinité » de clones.

Nous exposons, dans un premier temps, le patron de conception (Jaczynski 1998) permettant de masquer les opérations effectives de dulication à l'objet (le client) qui en effectue la requête.

Structure et constituants

Le patron `Composite` (fig. 3.5) permet de déléguer l'opération de réplication aux sous-classes.

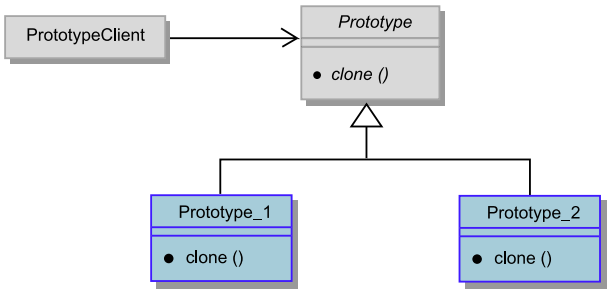


FIG. 3.5 – *Patron de conception permettant de **répliquer des objets** en masquant au client les opérations effectives de duplication.*

– **Prototype** définit l'interface de l'opération de réplication avec si besoin une implantation par défaut (*interface* ou *classe abstraite*).
– **Prototype_1** et **Prototype_2** implantent l'opération de réplication appropriée.
– **PrototypeClient** crée de nouveaux objets en demandant à un objet primaire de se répliquer.

Intérêts et limites

Le clonage d'un objet est encapsulé et le `client` n'a pas de connaissance explicite sur la classe de cet objet. Il ne manipule l'objet que du point de vue de l'interface `Prototype`. Cela permet par exemple de constituer des objets qui ne diffèrent que par des modifications faites incrémentalement. Le client transmet la requête de réplication, puis effectue les modifications incrémentales sur le clone.

Cependant, l'implantation de l'opération de clonage peut être difficile s'il existe des composants qui ne peuvent être clonés ou s'il y a des références circulaires (problèmes liés aux copies profondes). En particulier, la profondeur du clonage des composants à effectuer peut dépendre du contexte d'utilisation. La méthode `clone`, unique et sans paramètre, ne tient pas compte du contexte d'utilisation. De plus, la méthode de réplication doit prendre en compte l'ensemble des propriétés définies et héritées au niveau de la classe, ce qui pose des problèmes d'accès aux

propriétés privées (modifieur *private*) définies au niveau des *super-classes*. Enfin, lorsque l'objet cloné appartient à une hiérarchie souvent remise en cause et constituée d'un grand nombre de classes, il peut être très difficile de réécrire la méthode de réplication pour toutes les classes modifiées.

Il est donc intéressant de dissocier l'opération de création d'un nouvel objet de l'opération de recopie des propriétés, ce que nous proposons avec le patron `Prototype2`.

3.6.2 Réplication avec duplication déléguée

Structure et constituants

Le patron `Prototype2` (fig. 3.6) permet de répliquer un objet en déléguant la recopie des propriétés à l'objet nouvellement créé. La méthode `clone` s'occupe de créer une instance du même type, avec les propriétés non-initialisées. La méthode `copyProperty` de l'objet répliqué s'occupe de recopier les propriétés de l'objet original passé en paramètre. Celle-ci se charge de copier les propriétés locales à la classe en cours, puis délègue à son tour copie des propriétés héritées à la *super-classe* par l'appel récursif `super.copyProperties()`, qui remonte la hiérarchie de spécialisation en « collectant » les propriétés de la classe courante.

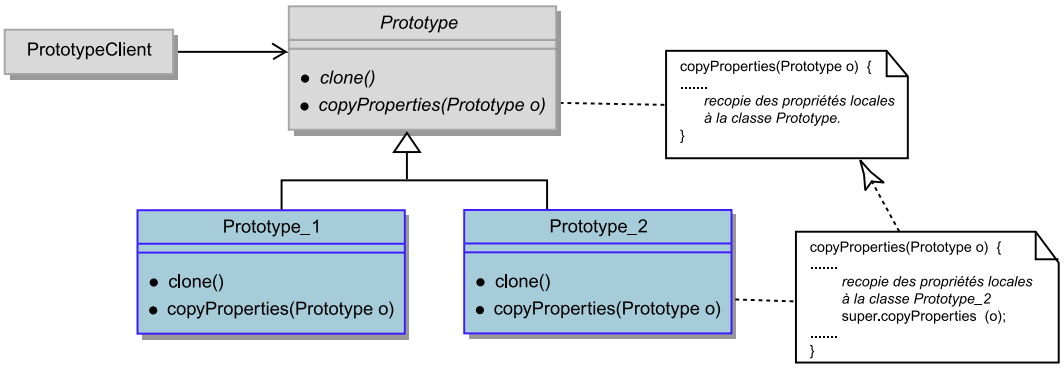


FIG. 3.6 – *Patron de conception permettant de* ***cloner des objets*** *avec déléguation de la recopie des propriétés.*

- `Prototype` définit l'interface de l'opération de clonage et de duplication des propriétés, avec si besoin une implantation par défaut.
- `Prototype_1` et `Prototype_2` implantent les opérations de clonage et de duplication appropriées.
- `PrototypeClient` crée de nouveaux objets en demandant à un objet primaire de se cloner.

Exemple d'utilisation du patron `Prototype2`

Considérons un exemple issu de l'astrophysique : *la classification des supernovae*. Une supernovae est une étoile qui explose : elle devient soudain des milliards de fois plus brillante, aussi brillante qu'une petite galaxie, mais ne vit au maximum que quelques semaines en changeant fréquemment de type. On peut les classer selon leurs caractéristiques spectroscopiques

(Bouquet 1998), qui permet de déterminer la présence de tel ou tel élément chimique (carbone, hydrogène, silicium, fer, etc.).

Imaginons que la classification des supernovae soit représentée par la hiérarchie de classes illustrée par la figure 3.7). La classe abstraite `ObjetStellaire` implémente l'interface `Prototype2`. Pour chaque type de supernovae est donc définie une méthode `clone()` et `copyProperties()`.

Soit un objet de la classe `TypeIb` dont on souhaite effectuer la réplication. Le diagramme de séquence (fig. 3.8) illustre le mécanisme de réplication mis en jeu par le patron `Prototype2`. Le mécanisme est le suivant :

1. Le client effectue la requête de réplication à l'objet `a` de type `TypeIb`,
2. `a` créé un nouvel objet `b` de même type,
3. `b` copie les propriétés de `a`, à l'aide de la méthode `copyProperties`, le long de la hiérarchie de spécialisation,
4. Le client reçoit l'objet `b` répliqué.

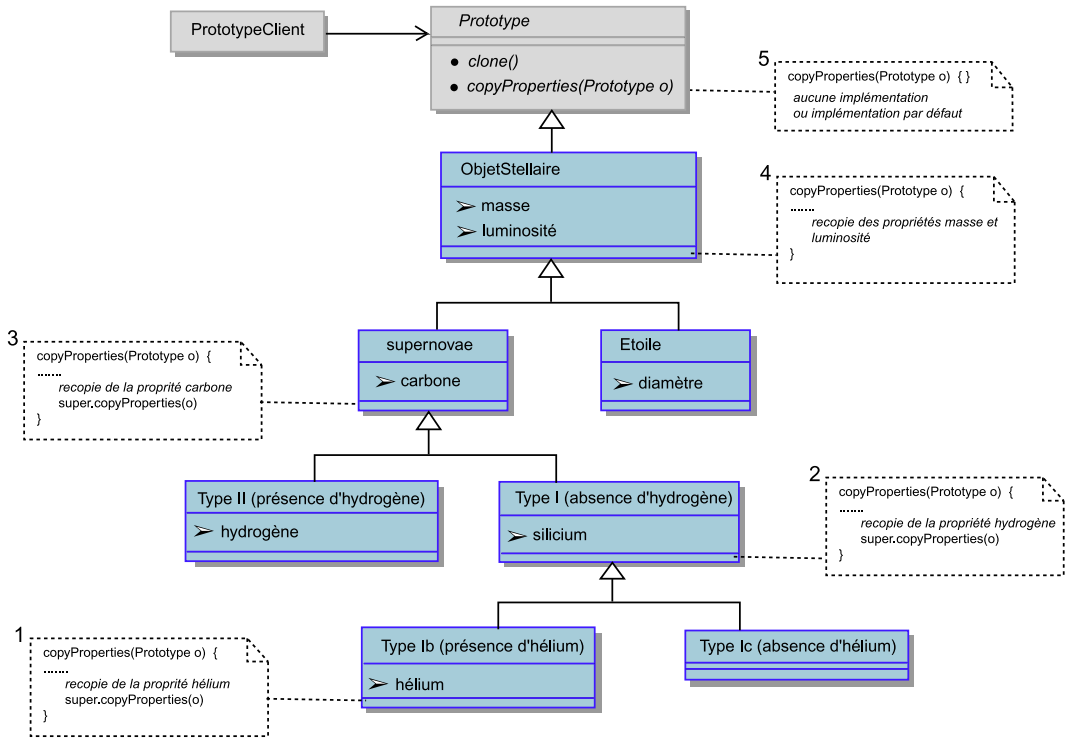


FIG. 3.7 – *Exemple d'application du patron* `Prototype2` *à la hiérarchie de classification des supernovae.*

Intérêts et limites

L'implantation de l'opération de clonage est simplifiée, car la recopie effective des propriétés est implantée au sein des classes qui définissent ces propriétés. Les problèmes de restrictions d'accès sont levés puisque les accès sont limités aux propriétés locales de la classe de définition. De plus, si une modification est effectuée au niveau de la hiérarchie des classes, les opérations de

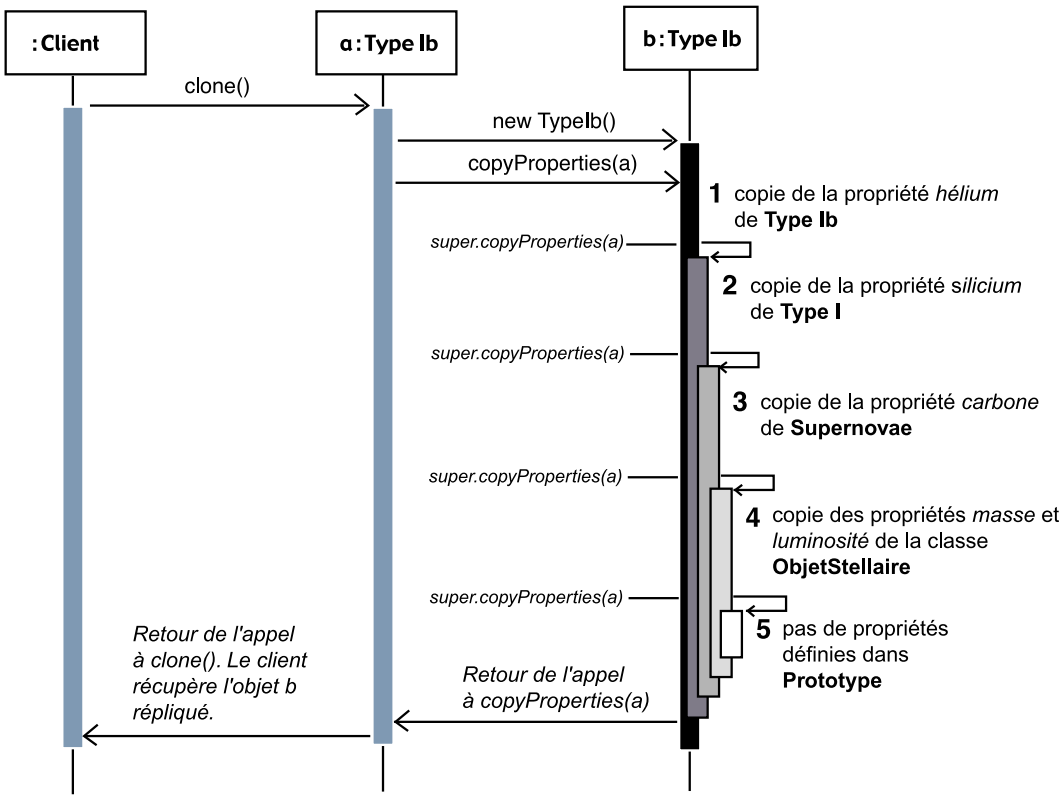


FIG. 3.8 – *Diagramme de séquence de réplication d'un objet par le patron* `Prototype2`

clonage et de recopie sont toujours valides.

Ce patron propose donc un mécanisme un peu plus efficace que le patron précédent, pour la réplication d'objet. Le patron suivant `Mutation` est une extension du patron `Prototype2` qui permet la mutation d'un objet, c'est-à-dire la transformation d'une instance de classe `A` vers une instance de classe `B`.

3.6.3 Mutation

Principe

Le principe de la mutation d'objet (fig. 3.9) que nous avons adopté peut être résumé de la manière suivante : pour pouvoir faire migrer une instance d'une classe source vers une classe cible, il faut tout d'abord savoir comment dupliquer l'objet, les propriétés de type simple (chaîne de caractères, entier, booléen, collections de type simple, etc.), les liens vers les autres objets, ainsi que les objets liés (cas d'une copie profonde). Ensuite, pour l'opération de mutation, il faut créer une nouvelle instance de la classe cible, puis recopier les propriétés partagées entre les deux classes, à partir de leur ancêtre commun.

Le modèle que nous proposons dans cette partie (fig. 3.9) offre un mécanisme générique pour « transformer » une instance d'un type `A` vers un type `B`. Le modèle est basé sur le patron

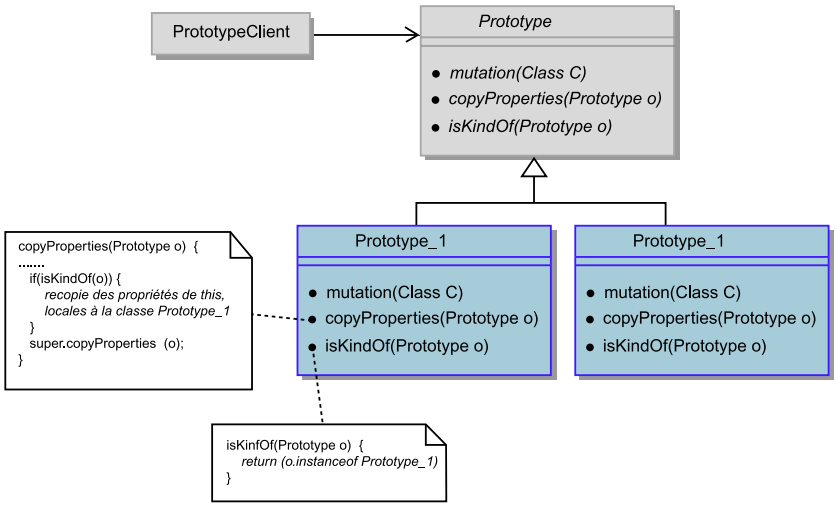


FIG. 3.9 – *Patron de conception pour la mutation d'objet*

`Prototype2` et diffère en deux points :

- l'appel à la méthode `clone()` de l'objet que l'on souhaite répliquer est substitué par l'appel à la méthode `mutation(Class C)`. Le paramètre C spécifie la classe cible de la mutation.
- un test de type (`isKindOf()`) est introduit au niveau de la méthode `copyProperties(Prototype a)`. La recopie des propriétés de la classe courante `C` n'est effectuée que si le type de l'objet `a` est un type valide : soit `a` est instance de `C` soit d'une sous-classe de `C`.

Exemple de duplication des propriétés

Un exemple de la structure de la méthode `copyProperties(Prototype o)` est donné par le pseudo-code objet Java suivant (cf. algorithme 3), implanté dans la classe `Supernovae`. Celle-ci doit être surchargée spécifiquement dans chacune des classes de la hiérarchie, et instanciée en fonction des variables présentes dans ces classes `Prototype` pour que le mécanisme reste valide. La fonction, récursive, remonte la hiérarchie des classes en copiant à chaque étape les propriétés locales. L'enchaînement des appels se termine à la racine de la hiérarchie.

Algorithme 3 Exemple de duplication déléguée des propriétés par utilisation de la hiérarchie

```
Float : carbone [Intensité du spectre lumineux associée à la présence du carbone]
void : copyProperties (Prototype : o) {
si this.isKindOf (o) alors
  this.carbone ← o.carbone
fin si
super.copyProperties(o) [Appel récursif en remontant la hiérarchie]
}
```

Usage de l'introspection

Il est cependant assez contraignant d'avoir à surcharger dans chacune des classes de la hiérarchie, une méthode qui a la même structure et ne diffère que par les propriétés manipulées. Une solution permettant de résoudre cette contrainte consiste à accéder à la classe associée à l'objet, afin de l'interroger sur les propriétés qui y sont déclarées. Cette opération, possible en Java, est appelée *introspection*.

Notons qu'il existe un certain nombre de langages à objets, dits *réflexifs*, tels que Smalltalk ou Ptitloo (Ferber 1990) par exemple, qui offrent la possibilité de manipuler les classes par l'intermédiaire de classes particulières, appelées *méta-classes* (classes de classes). Les classes sont alors vues comme des objets, chacun d'eux étant instance d'une méta-classe. La réflexivité permet ainsi de modifier la sémantique des classes usuelles. L'introspection est un mécanisme plus simple, qui ne permet pas de modifier les classes (pendant l'exécution), mais simplement d'accéder à la classe associée à un objet pour lire les propriétés et les méthodes qui y sont définies.

Soit o une instance, $o.getClass()$ retourne la Classe associée à o, c'est-à-dire un objet de type `Class`. Soit c cet objet, $c.getFields()$ retourne la collection des variables déclarées dans c, $c.getMethods()$, la liste des méthodes. La méthode `copyProperties` peut donc s'écrire une bonne fois pour toute à la racine de la hiérarchie, à un ordre supérieur [75] simplement de la façon suivante (algorithme 4) :

Algorithme 4 Duplication des propriétés, approche itérative en utilisant l'introspection

```
void : copyProperties (Prototype : o) {
{ Field } : f1 ← this.getClass().getFields() [Liste des propriétés de l'instance répliquée (this)]
{ Field } : f2 ← o.getClass().getFields() [Liste des propriétés de l'instance d'origine]
pour tout élément e de f1 faire
  si e ∈ f2 alors
    this.e ← o.e
  fin si
fin pour
```

Exemple d'utilisation du patron `Mutation`

La figure 3.10 illustre la mutation de l'étoile Bételgeuse en une supernovæde type `TypeIc`. L'algorithme 3 est utilisé pour la duplication. Notons qu'à la fin du processus de mutation, l'objet initial est détruit. Cette étape n'est pas intégrée au patron de conception `Mutation`, on peut choisir ou non de conserver l'instance initiale.

75. Il existe une certaine analogie entre les *fonctions introspectives* des langages objets et les fonctions d'*ordre supérieur* des langages fonctionnels.

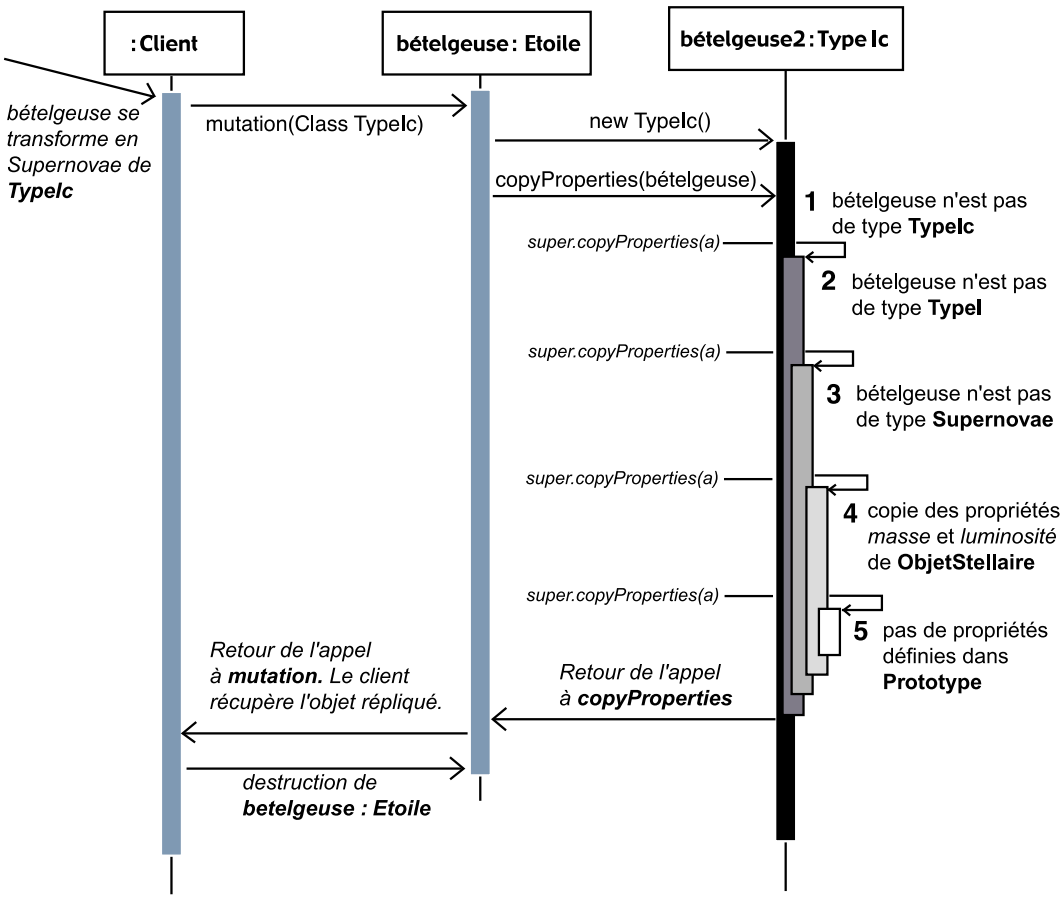


FIG. 3.10 – *Diagramme de séquence illustrant les différentes étapes de mutation d'un objet.*

Intérêts et limites

Le patron de conception `Mutation` offre un modèle générique permettant de simuler l'évolution de la caractéristique la plus fondamentale des objets : sa classe d'appartenance sous-tendus par le lien is-a. Il offre de plus les avantages des patrons décrits précédemment : l'encapsulation de l'opération de réplication et l'incrémentalité dans la création des objets.

Il est cependant nécessaire, après le mutation, de définir toutes les propriétés du nouvel objet qui sont inexistantes dans l'objet d'origine. Elles se trouvent en effet affectées de la valeur non-définie, c'est-à-dire le pointeur `null`.

Notre dernière proposition concernant la gestion de l'évolution du type des objets concerne une manière pratique de compléter l'information manquante dans le nouvel objet, par un mécanisme d'hybridation. Le mécanisme d'hybridation est exposé par le patron de conception suivant.

3.6.4 Hybridation

Le patron *Hybridation* (fig. 3.11) offre un mécanisme générique pour « transformer » une instance `a` d'un type `A` vers un type `B`, en la croisant avec une instance `h` du même type que `B`. Cette opération de croisement permet à l'objet muté de compléter ses propriétés manquantes,

par celles de `h`. Ce modèle est une extension du patron Mutation (§ 3.6.3).

- la méthode `mutation(Class C)` est substituée par la méthode `hybridation(Class C, Prototype h)`. Le paramètre `C` spécifie la classe cible de la mutation. `h` est l'objet avec lequel on souhaite effectuer l'hybridation et doit être une instance de la classe `C`.
- la méthode `copyProperties` est étendue à deux paramètres correspondant à l'objet d'origine et l'objet `h`, respectivement.

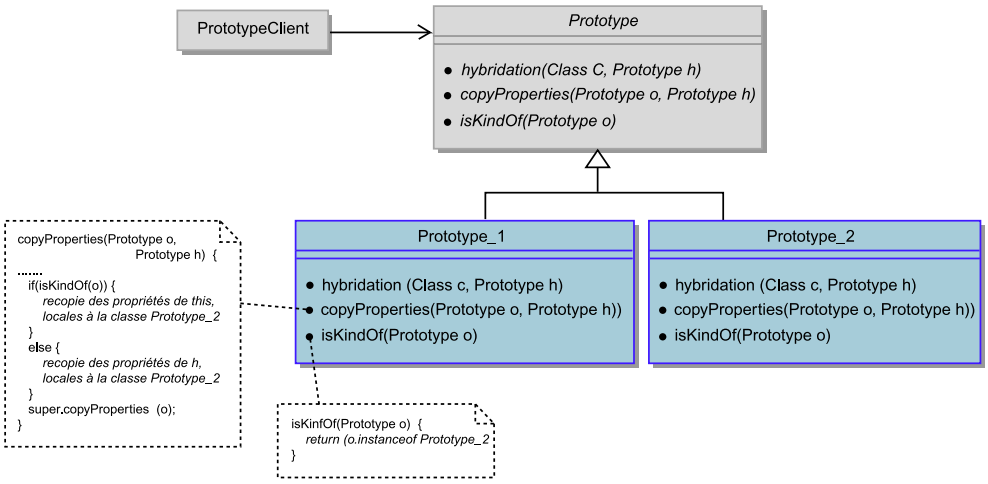


FIG. 3.11 – *Patron de conception pour l'hybridation d'objet*

Exemple d'utilisation du patron `Hybridation`

La figure 3.12 illustre l'hybridation de l'étoile Bételgeuse avec un autre objet `h`. Le résultat de l'hybridation est une instance de `TypeIc` avec les mêmes propriétés que h, sauf pour les propriétés communes, déjà définies dans l'objet d'origine.

3.6.5 Conclusion sur l'évolution

Nous avons exposé différents mécanismes sous la forme de patrons de conception, utilisables dans la plupart des langages à objets, permettant de gérer d'une manière efficace l'évolution d'objet (réplication, mutation et hybridation) de manière transparente. Ces mécanismes sont appliqués et largement utilisés au sein de la plate-forme $\mathcal{IKBS}$, comme nous l'illustrerons au chapitre 6.

Ces patrons de conception consituent la base du mécanisme plus général de gestion de l'évolution des connaissances pour la mise en oeuvre de l'itération.

3.7 Conclusion

Nous avons présenté un ensemble de mécanismes internes aux éléments constitutifs du modèle CoDesc (modèles dscriptifs, composants, attributs, descriptions). Ces mécanismes internes sont

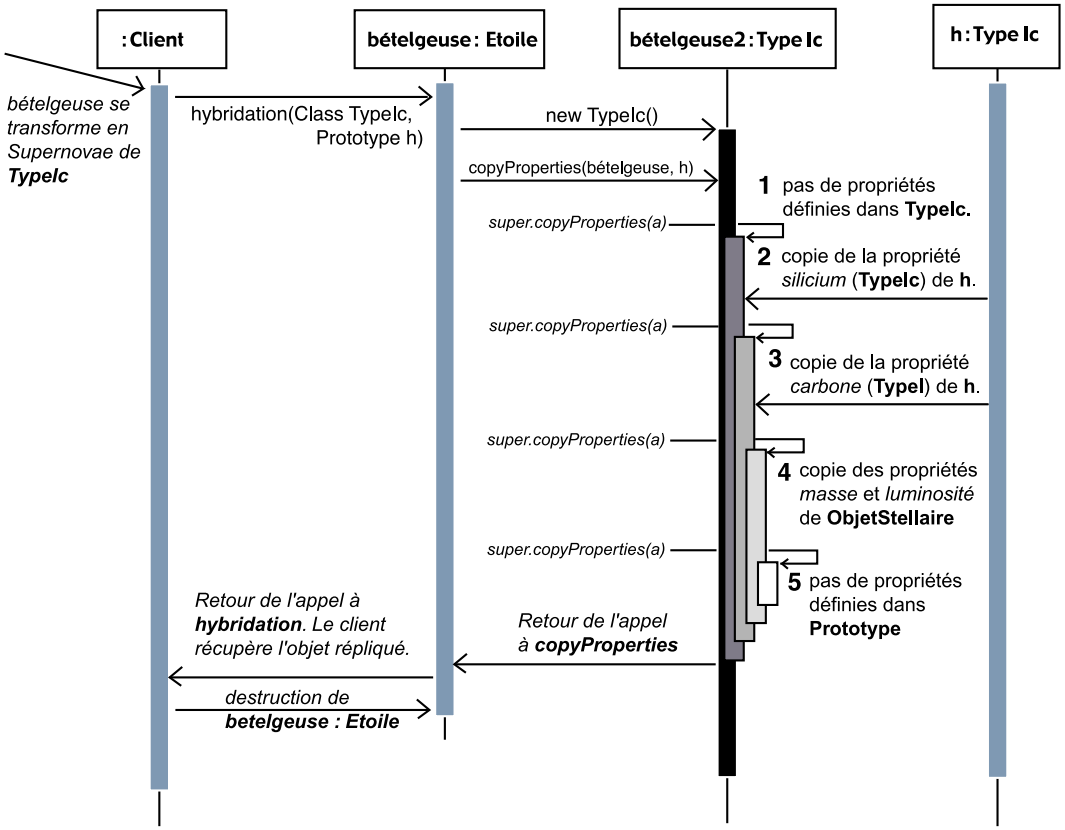


FIG. 3.12 – *Diagramme de séquence illustrant les différentes étapes de l'hybridation de objet* ***bételgeuse :Etoile*** *avec l'objet* ***h :TypeIc****.*

très utiles car ils offrent une base solide pour le développement de méthodes d'analyse et de gestion des connaissances de plus haut niveau.

En particulier, le mécanisme de gestion de l'évolution est à notre sens très original car il est générique et applicable à toute hiérarchie de classes d'un langage de programmation orienté objet, lorsqu'il est nécessaire de simuler l'évolution des objets.

Par ailleurs, le modèle CoDesc dispose de fonctionnalités internes que nous n'avons pas exposées dans le cadre de cette thèse. En particulier, la synthèse de nouveaux concepts à partir d'un ensemble de descriptions ainsi que les opérations de filtrage permettant de sélectionner un sous-ensemble d'individus de la base de connaissances correspondant aux critères exprimés par le filtre. Ce mécanisme de filtrage est sensiblement équivalent aux requètes des bases de données.

Troisième partie

Identification, comparaison et classification d'objets complexes

Chapitre 4

Identification d'objets complexes

Sommaire

4.1 Introduction

Identifier consiste à associer un objet inconnu ou un nouvel évènement, à un concept préexistant. C'est par exemple l'action effectuée par un médecin lors d'un diagnostic : à partir d'un ensemble de symptômes observés sur un patient, il associe une maladie. C'est également ce que font les écologistes, les systématiciens ou les observateurs de la nature lorsqu'ils cherchent à reconnaître le taxon (la famille, le genre, l'espèce, etc.) auquel appartient un spécimen découvert. Pouvoir nommer un objet, c'est un peu mieux le connaître, et cela présente l'avantage, parfois indispensable, de déduire certaines propriétés utiles sur l'objet. Par exemple que le champignon que l'on vient de cueillir n'est pas comestible... D'une façon générale, le positionnement de l'objet étudié dans la Classification [76] est indispensable à un grand nombre de travaux en biologie (épidémiologie, génétique, écologie, etc.).

L'identification précise des objets de la nature est cependant un problème souvent difficile. Comme le souligne Jacques Lebbe (1991), associer un spécimen découvert à un nom d'espèce est sans doute un des plus important problème auquel les scientifiques se trouvent confrontés, en raison du nombre considérable des concepts à prendre en compte (plusieurs millions). Bien sûr, dans la pratique, le cadre de l'étude se limite à un sous-ensemble du vivant bien déterminé, et l'identification consiste alors à affiner le *classement* de l'objet dans les hiérarchies préexistantes. Toute identification nécessite une hypothèse sur l'objet d'étude.

Pour notre étude sur les coraux, le concept de plus haut niveau considéré est fixé à l'Ordre des *Scléractiniaires* (coraux durs), qui regroupe 13 familles, 55 genres et plusieurs centaines d'espèces dans les Mascareignes. L'identification est difficile dans ce domaine, car les coraux sont des objets *polymorphes*. La variabilité intra-spécifique [77] et même intra-coloniale [78] peut être parfois très importante. Une connaissance minimale de la systématique du domaine ainsi qu'un bon sens de l'observation sont souvent pré-requis.

Pour certaines branches du vivant, le verdict final d'une identification, le nom de genre ou d'espèce, est du ressort d'un expert du domaine. Elle est parfois accompagnée d'un certain degré d'incertitude.

Ce chapitre se situe dans le cadre de l'Identification Assistée par Ordinateur (I.A.O.) pour la biologie et par extension, à la médecine, pour l'identification des pathologies. Nous présen-

76. la Classification des êtres vivants est la hiérarchie linéenne constituée des taxons.
77. au sein d'une même espèce.
78. Les coraux sont formés pour la plupart de colonies d'organismes microscopiques, les *polypes*. La famille des Fungidae est la seule dont la « colonie » est formée d'un polype solitaire qui peut atteindre plusieurs centimètres.

tons tout d'abord un rapide aperçu des méthodes utilisées depuis l'époque ou les scientifiques se sont intéressés à décrire la nature et à formaliser leur approche. Nous abordons ensuite le fonctionnement des systèmes informatisés d'aide à l'identification, développés très tôt par des biologistes, précurseurs dans ce domaine, ainsi que quelques méthodes d'identification issues des recherches en Analyse de Données et Intelligence Artificielle. Nous exposerons ensuite la méthode développée au sein du système $\mathcal{IKBS}$, qui travaille à partir de données exprimées dans le formalisme CoDesc (chap. 2). La méthode complète est fondée sur une combinaison de méthodes *monothétique* et *polythétique*, qui prend en compte la complexité des descriptions, ainsi que les connaissances relatives au domaine, les « connaissances de fond ». Enfin, nous développerons une application à l'identification de spécimens coralliens et tenterons de montrer l'adéquation de la méthode à la problématique : identifier des objets complexes

4.2 L'identification en biologie

Dans le but d'identifier un spécimen, deux approches sont généralement adoptées par les experts : la méthode *synthétique* et la méthode *analytique* (Lebbe 1991).

4.2.1 Méthode synthétique et méthode analytique

La méthode synthétique identifie un objet inconnu par observation globale, sans en détailler les caractéristiques. C'est une forme de raisonnement par analogie fondée essentiellement sur l'intuition et l'expérience de l'expert. L'homme possède en effet de grandes facultés de reconnaissance synthétique, une capacité à appréhender la réalité globalement et à saisir en un court instant un grand nombre de paramètres. Il lui est malheureusement souvent difficile d'analyser et de retransmettre ses facultés d'identification à autrui et à expliquer les mécanismes de raisonnement.

Les méthodes analytiques ont été inventées pour pallier à cette difficulté. Elles sont fondées sur la comparaison de la description de l'individu à identifier aux représentations des différents concepts auxquels l'individu peut appartenir.

Les experts adoptent donc une combinaison des méthodes synthétique et analytique, grâce à certaines heuristiques acquises par expérience. Ils savent très rapidement situer le spécimen dans un groupe taxonomique particulier par observation globale des caractères (sans les détailler explicitement), puis analyser certains d'entre eux plus particulièrement pour confirmer ou infirmer leur intuition en s'appuyant sur la littérature (essentiellement les monographies).

La méthode analytique

Deux variantes de la méthode analytique sont possibles : la **discrimination** qui est monothétique, et la **correspondance** (ou matching en anglais) qui est polythétique.

- **La discrimination** (où l'élimination) consiste à observer les caractères du spécimen étudié l'un après l'autre et à éliminer les taxons qui ne correspondent pas aux valeurs observées. Cette méthode se fonde généralement sur la construction de **clefs d'identification** (ou clefs diagnostiques).

– **La correspondance** implique une comparaison directe du spécimen avec la description des taxons. La comparaison est généralement effectuée à l'aide d'une mesure de ressemblance (ou similarité) correspondant aux nombres de caractères similaires (ou dissimilaires) entre le spécimen étudié et les descriptions de concepts. Cette façon de procéder offre la liberté de décrire les caractères dans un ordre quelconque, ce qui implique une certaine connaissance du domaine (choix des caractères pertinents et fiables par exemple).

Les clefs d'identification

La création de clefs d'identification [79] est, depuis le XVIIIème siècle (Lamarck, 1778) [80] une part importante du travail des systématiciens (Lebbe 1991, Vignes 1996). C'est une tâche longue et délicate qui nécessite une très bonne connaissance et une large expérience du groupe zoologique considéré. Le systématicien doit étudier un grand nombre d'échantillons, ainsi que les types (prototypes historiques stockés dans les collections des Muséums), les comparer avec d'autres spécimens de familles assez proches, sélectionner les critères les moins ambiguës et les plus pertinents et enfin construire une liste ordonnée de critères. L'ensemble constitue une clef pour un groupe zoologique particulier. Certaines clefs peuvent faire intervenir un nombre considérable de variables, comme la clef de Thonner, revue en 1981 pour identifier les familles végétales, qui comprend 2117 Nœuds (Vignes 1991), ou encore se restreindre à un petit groupe taxonomique particulier (Coste 1980). Tout l'art consiste à trouver et ordonner les bons caractères, ce qui est l'affaire de spécialistes connus et réputés dans le domaine. La difficulté est d'autant plus importante, que les clefs peuvent être remises en question, dès lors que l'on dispose de nouvelles connaissances du domaine. La difficulté de ce travail de synthèse explique que les premières méthodes automatiques d'identification furent très tôt appliquées à la systématique (Belson 1959).

L'apport de l'informatique

La connaissance taxonomique ne se résume pas simplement à la construction de clefs d'identification ou même de hiérarchies de taxons (la classification). C'est une théorie très complexe qui émerge de l'étude des caractères, de leur variabilité, de leur signification phylogénétique [81] et écologique, etc. Dans un grand nombre de groupes zoologiques, la connaissance incomplète ne permet pas de construire une théorie définitive du domaine. C'est le cas par exemple des éponges marines ou des coraux. En effet, les experts ne s'accordent pas sur le nombre d'espèces de coraux présentes sur la planète, qui oscille entre 500 et 900 espèces (Faure 1982, Veron 2000).

Un facteur important permettant d'établir une identification correcte passe par la définition de caractères descriptifs permettant de séparer de façon non ambiguë les taxons. Ceux qui permettent d'affirmer l'appartenance du spécimen au taxon sont appelés *caractères diagnostiques*. Ces caractères doivent être stables, c'est-à-dire indépendants des facteurs extérieurs tels que le milieu écologique. En effet, l'influence de l'environnement peut parfois avoir une grande incidence sur la morphologie. Pour les coraux par exemple, le milieu influence fortement la forme générale

79. appelées également clefs de détermination ou clefs diagnostiques.

80. La première clef publiée se trouve dans la « Flore françoise » de Jean-Baptiste de Monnet, chevalier de Lamarck.

81. La **phylogénie** est le point de vue de la classification qui s'intéresse aux relations de parenté existant entre les êtres vivants.

de la colonie : selon la profondeur de localisation, le régime de marée et de houle, la colonie adopte une forme plus ou moins branchue pour être plus résistante ou pour au contraire, capter plus de lumière. Une étude statistique devrait donc être conduite pour connaître l'incidence des facteurs environnementaux sur les caractères avant de les considérer comme caractères diagnostiques. Dans de nombreux domaines taxonomiques, ce type d'étude est loin d'être achevé.

L'informatique peut se révéler d'une grande aide, dans cette discipline où les données sont nombreuses et les concepts difficiles à décrire. En effet, les clefs d'identification donnent le moyen de réaliser des identifications, mais la liste d'attributs et l'ordre dans lequel ils devront être utilisés pour décrire l'individu à identifier sont imposés par la constitution de la clef. Si l'utilisateur n'a pas accès à l'attribut de l'individu pour un de ces descripteurs, l'identification ne peut aboutir. L'Identification Assistée par Ordinateur (I.A.O.) a pour objectif de proposer des méthodes d'identification qui soient interactives et qui s'adaptent aux différentes circonstances d'identification (présence de valeurs manquantes, imprécises, multiples, etc.).

4.2.2 L'Identification Assistée par Ordinateur (I.A.O.)

Très tôt, dès le début des années 60, un grand nombre de systèmes d'aide à l'identification ont été proposés pour l'identification de spécimens biologiques. Ces programmes s'appuient sur le principe des clefs dichotomiques et travaillent sur des tableaux de données (Hall 1970, Pankhurst 1970, Dallwitz 1974, Payne 1975). Pour un historique des méthodes d'identification, voir par exemple (Pankhurst 1998). Ces tableaux se présentent sous la forme d'une liste de taxons caractérisés par des valeurs simples, essentiellement numériques. Ces tableaux doivent être complets, ce qui implique l'observation de tous les caractères pour tout taxon, ce qui n'est pas le cas lors de la constitution de clefs manuelles. Ce type de représentation ne permet pas de gérer l'absence de valeur, la variation, l'imprécision, et l'inapplicabilité des descripteurs.

Pour combler les lacunes liées aux représentations tabulaires, le langage DELTA [82] (Dallwitz 1978) a été créé pour la représentation des connaissances en systématique.

DELTA

DELTA est adopté comme le langage de référence pour la représentation des données taxonomiques, par l'International Taxonomic Databases Working Group (TDWG). De nombreux programmes ont été développés autour de ce formalisme, par exemple CONFOR (Dallwitz 1984), PANKEY (Pankhurst 1986) et en particulier INTKEY développé par les créateur de DELTA qui permet de construire des clefs interactives. DELTA a été mis à jour régulièrement (Dallwitz 1993) depuis sa création, ce qui souligne l'importance de ce langage de représentation des descriptions pour la communauté des systématiciens.

Voici une liste des fonctionnalités principales des systèmes construits autour de DELTA qui illustre la richesse du langage et met en évidence les besoins des systématiciens :

– Le langage DELTA est un format d'écriture de texte permettant d'expliciter les noms des taxons, de définir des descripteurs et leurs domaines de valeurs, et d'exprimer des relations

82. DELTA signifie DEscription Language for TAxonomy

de dépendance entre descripteurs. Il permet également de représenter les descriptions, en relation avec les concepts.

- Les outils développés autour de DELTA sont nombreux et témoignent de l'adéquation du langage aux besoins exprimés par les systématiciens. CONFOR convertit le format DELTA en langage naturel, Le programme KEY génère des clefs d'identification conventionnelles et INTKEY, des clés dynamiques. En sélectionnant un caractère, le programme détermine son pouvoir discriminant. Un poids peut être associé à chaque caractère pour tenir compte de la facilité d'observation de ce caractère et paramétrer les clefs d'identification.
- Même s'ils ne travaillent pas directement sur le format DELTA, de nombreux programmes dédiés à la systématique importent ou exportent des données dans ce format. $\mathcal{IKBS}$ possède également un module d'importation du format DELTA vers le modèle de représentation CoDesc, qui a été réalisé à l'aide d'outils d'analyse lexicale et syntaxique développé en Java : JAVALEX et JAVAYACC. La mise au point du module d'exportation permettrait d'offrir à la communauté scientifique les descriptions de concepts et de spécimens au format DELTA.

4.3 L'identification du côté de l'Analyse de Données et de l'Intelligence Artificielle

Dans le domaine de l'Analyse de Données et de l'Intelligence Artificielle, les méthodes d'identification sont très nombreuses. Elles sont appelées *méthodes de discrimination* en analyse discriminante (Caraux & Lechevallier 1996), *méthodes de classements* ou encore *classification d'instances* dans les représentation à objets. En Analyse de Données, les graphes d'identification sont également appelés arbres de segmentation (Lambert & Williams 1966, Hall 1970). Dans les années 80, les graphes d'identification ont trouvé leur application dans le domaine de l'apprentissage automatique où ils sont désignés sous le nom d'**arbre de décision**. Les travaux de Breiman et de Quinlan ont donné naissance aux systèmes CART (Breiman et al. 1984) et ID3 (Quinlan 1986) et ses successeur C4.5 (Quinlan 1992) et C5.0. Durant ces quinze dernières années, de nombreux travaux ont permis d'apporter des améliorations à ces algorithmes, comme par exemple les techniques d'élagage, qui visent à réduire la taille de l'arbre construit.

4.3.1 Les arbres de décision

La méthode CART (Breiman et al. 1984) construit des arbres de décision binaires (fig. 4.1) à partir d'un ensemble $O = \{X_1, \ldots, X_n\}$ d'individus appelé ensemble d'apprentissage. L'ensemble O est muni d'une partition en k classes $C_1, \ldots, C_k$, définies *a priori*. Les individus sont décrits par un ensemble $A = \{A_1, \ldots, A_p\}$ d'attributs. L'ensemble des valeurs qui peuvent être associées à l'attribut A_i est appelé le domaine de A_i et est noté D_i.

Un arbre de décision est un graphe défini par un couple (S, A_r) où S est l'ensemble des *sommets* de l'arbre et A_r est un sous-ensemble de $S \times S$ dont les éléments sont les *arcs* de l'arbre. Chaque arc se présente donc comme un couple (x,y) où $x \in S$ est l'origine de l'arc et $y \in S$ est l'extrémité. Chaque sommet non-terminal de l'arbre de décision, appelé nœud de décision, est associé à une *question binaire* et chaque nœud terminal (les feuilles) est associé à

une classe C_i.

L'arbre de la figure 4.1 a été obtenu à partir d'un ensemble d'individus muni d'une partition en deux classes, *malade* et *bonne santé*. Deux attributs interviennent pour les nœuds de décision, la *température*, qui est un attribut continu, de domaine [36 43] et l'attribut *maux de gorge* qui est binaire de domaine {oui, non}.

L'arbre classifie la description des symptômes d'un individu $X \in O$ à travers une chaîne de questions binaires, partant de la racine vers les feuilles en choisissant à chaque étape une des deux réponses possibles. Lorsqu'une feuille est rencontrée la classe correspondante est associée à X.

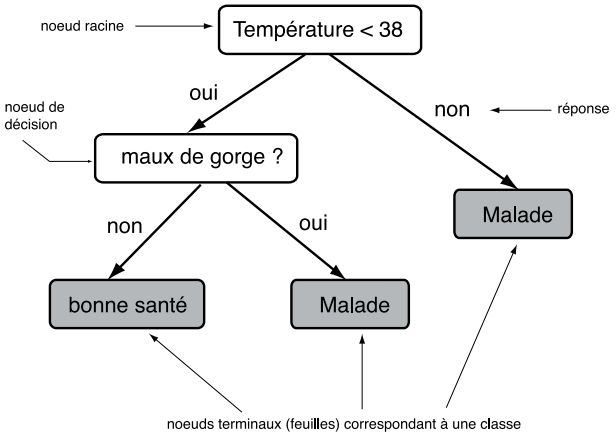


FIG. 4.1 – *Exemple d'arbre de décision binaire.*

Un arbre de décision peut également être utilisé pour extraire des *règles de production*, appelées également *règles de décision* ou *règles de classification*.

En effet, chaque chemin maximale de l'arbre peut en effet être interprété comme une règle dont la conclusion est la classe étiquetant la feuille correspondante. L'ensemble des règles suivantes peut-être extrait de l'exemple précédent :

```
SI [Température < 38 = oui] et [maux de gorge = non] ALORS Bonne santé
SI [Température < 38 = oui] et [maux de gorge = oui] ALORS Malade
SI [Température < 38 = non] ALORS Malade
```

La possibilité d'extraire des règles de décision d'un arbre de décision explique le fort pouvoir explicatif de cette approche et son succès en Intelligence Artificielle. Les règles sont simple à interpréter (quand les chemins de la racine vers les feuilles sont minimisés) et s'expriment dans le même langage de description que les données de départ.

La méthode de construction monothétique que nous proposons par la suite étant une extension des arbres de décision, nous en donnons ci-dessous le principe général de construction.

Principes de construction d'un arbre de décision

Classiquement, un arbre de décision est construit par l'application récursive de trois fonctions essentielles :

1. *Fonction d'arrêt.* La fonction d'arrêt permet de tester si un noeud est terminal. Si c'est le cas, ce noeud est transformé en feuille et on lui attribue la classe la plus probable parmi les cas qu'il contient. La probabilité est évaluée au travers de la fréquence relative de chaque classe associée aux cas présents dans le noeud.
2. *Fonction de sélection.* Cette fonction recherche le meilleur descripteur permettant de créer une partition optimale des cas vis-à-vis d'un descripteur cible.
3. *Fonction de partitionnement.* L'ensemble des cas présents pour le noeud courant est découpé en sous-ensembles disjoints, en fonction de leur valeur pour le descripteur choisi.

Une fois construit, la complexité de l'arbre peut être diminuée par différentes techniques, appelées « élagage », mettant en oeuvre des heuristiques (ou connaissances stratégiques) ou utilisant des propriétés statistiques des noeuds de l'arbre.

La qualité d'un arbre dépend essentiellement du choix de bons descripteurs dans les noeuds internes. Ce point de vue est souvent controversé. Certaines conclusions de Breiman et al. (1984) et Mingers (1989) tendent à prouver que le choix des descripteurs est sans influence sur la qualité de l'arbre, seule la méthode d'élagage étant importante. Cela peut être un point de vue validé, dans la mesure où chaque descripteur est considéré équivalent aux autres, mais dans les applications réelles, les descripteurs ont des importances relatives. Certains sont non signifiants ou bruités, les autres sont connus pour être particulièrement importants. Le problème est que parfois l'ensemble d'apprentissage ne retransmet que rarement cette réalité. Ces considérations ne sont que rarement prises en compte dans le domaine de l'apprentissage à partir d'exemples. L'Intelligence Artificielle a le mérite d'introduire des heuristiques, des connaissances stratégiques, afin de rendre compte au mieux de la réalité.

4.3.2 Quelques outils de construction de clés d'identification

4.3.3 MAKEY

L'algorithme MAKEY (Lebbe 1991, Vignes 1991) est une extension des arbres de décision pour l'exploitation de données symboliques. Il a été utilisé dans de nombreuses applications en biologie et en médecine (Vignes 2000), en particulier pour l'identification des *phlébotomes américaines* (Diptères *Psychodidae*) qui compte environ 400 espèces.

Données considérées

Les données employées sont des descriptions de concepts, caractérisées par des attributs qualitatifs disjonctifs, c'est-à-dire que chaque attribut est caractérisé par une disjonction de valeurs qui représente les valeurs admissibles pour un membre du concept. Les classes utilisées pour construire le graphe d'identification [83] ne sont plus, comme dans le cas classique, décrites

83. MAKEY considère des graphes acycliques orientés, chaque classe est représentée par une unique feuille.

en extension par énumération de leurs membres, mais en *intension* par un objet symbolique. L'algorithme tient compte de relations logiques entre attributs : l'absence de certaines valeurs peut s'expliquer soit par manque d'information, soit par impossibilité logique de décrire cet attribut. Les relations logiques entre attributs expliquent les valeurs dites **inapplicables**. Par exemple, l'absence des ailes pour un oiseau implique l'inapplicabilité de l'attribut *couleur des ailes*. Ceci permet aux descriptions de garder une structure cohérente sans exiger que tous les concepts utilisent tous les attributs.

Principe de l'algorithme

Lors du développement du graphe d'identification, la sélection d'un attribut se fait suivant son pouvoir discriminant, qui mesure la capacité de l'attribut à discriminer les concepts (Vignes 1991). Les relations entre attributs sont prises en compte pour le calcul du pouvoir discriminant d'un attribut de la façon suivante : si $a \Rightarrow b \Rightarrow c$ le pouvoir discriminant de a est représenté comme le *maximum* des pouvoirs discriminants de a, b et c.

Plus récemment MAKEY a été étendu afin de gérer des descriptions plus complexes (Vignes 2000). Ainsi, des valeurs munies de typicité sont introduites dans les descriptions, afin de permettre d'ordonner les valeurs d'un descripteur pour un taxon selon leur fréquence relative les unes par rapport aux autres (Lebbe & Vignes 1993). La notion d'incertitude exprimée par les experts a été introduite par l'intermédiaire de *descriptions individuelles probabilistes* (Ciampi et al. 1994, Ciampi et al. 1996). La prise en compte de ce type de descriptions pour la construction du graphe d'identification consiste à répartir chaque individu dans plusieurs parties du graphe, selon une certaine *masse* (Perinel 1996).

De plus une certaine connaissance stratégique est introduite sous la forme de préordres associés aux descripteurs et aux concepts, ce qui a pour effet d'orienter la stratégie de construction des arbres de discrimination selon des directives (connaissances) imposées par l'utilisateur (Vignes 2000).

4.3.4 AlFReDO

L'algorithme `AlFReDO` (Simon & Napoli 1997, Simon 2000) est un bon exemple d'extension des méthodes classiques de construction d'arbres de décision adaptée à une représentation des connaissances par objets (RCO). Les données sont des instances de classes qui appartiennent à une hiérarchie. Les classes cibles sont définies par l'utilisateur avant la création de l'arbre, et peuvent être organisées en hiérarchies de classes virtuelles correspondant à un point de vue particulier. `AlFReDO` prend en compte les attributs mono ou multi-valués, et les relations mono ou multi-valuées. Il peut également importer des tableaux de données classiques.

4.4 Identification d'objets complexes, l'approche IKBS

La méthode que nos proposons est une méthode d'identification monothétique, qui construit des arbres d'identification à partir de connaissances représentées dans le formalisme CoDesc (cf. §2). Cette méthode a été présentée dans Conruyt & Grosser (1997) et étendue dans Grosser

& Conruyt (1999), nous en donnons ici les principaux aspects.

Les connaissances considérées sont de deux types : des descriptions de spécimens et des représentations des différents concepts.

Les descriptions sont des connaissances observées, structurées et munies de valeurs complexes (ensembles, conjonctions, inconnues, intervalles, etc.). Elles sont généralement le résultat d'observations d'échantillons réels, décrites minutieusement par les experts. Elles peuvent également être des représentations fictives, modélisées pour exprimer la typicité d'une espèce ou d'un taxon particulier. C'est souvent une solution adoptée par défaut, lorsque l'expert ne dispose pas d'échantillons suffisamment représentatifs du taxon considéré. Enfin, les descriptions peuvent être le résultat de méthodes de synthèse appliquées sur un ensemble de descriptions dans le but d'extraire une représentation prototypique d'un taxon. Cette opération de synthèse permet en particulier de simplifier l'espace d'apprentissage en réduisant l'ensemble des observations à prendre en compte par l'algorithme d'identification.

D'autre part, les descriptions sont des instances de concepts, organisés en hiérarchie de spécialisation, qui modélisent des « connaissances de fond » (Background knowledge) relatives au domaine abordé par les données. Cette modélisation des connaissances de fond permet d'exprimer des propriétés particulières des concepts, par l'intermédiaire de relations de dépendance entre descripteurs (connaissances structurelles), de règles logiques permettant d'effectuer des inférences afin de mettre en oeuvre certaines « connaissances stratégiques » ou encore en qualifiant l'importance relative des descripteurs par l'intermédiaire de poids. Les concepts peuvent également être interprétés comme des descriptions *en intension* des taxons de plus haut niveau.

4.4.1 Objectifs

A partir des descriptions pré-classifiées attachées aux différents concepts modélisés, la méthode proposée a pour objectif de construire une procédure de classification représentée par un *arbre d'identification* [84]

La méthode doit tenir compte des deux niveaux de classement des descriptions :

- Au niveau des concepts. Chaque description est instance d'un concept modélisé, qui lui donne sa structure et un ensemble de descripteurs permettant de le décrire. Les concepts sont organisés en une hiérarchie de concepts (cf. 2.6) qui correspond généralement, en systématique, aux taxons de plus haut niveau (Ordre, Famille ou Genre) du domaine modélisé.
- Au niveau des classes. La donnée d'un descripteur particulier, l'*attribut de classe* noté a_c, définit l'attribut cible de l'identification. Les différentes classes correspondent aux valeurs du domaine attaché à l'attribut, qui lorsqu'il est de type hiérarchique, définit une *taxonomie locale*. En systématique, cette taxonomie modélise généralement les taxons de plus bas niveau, les genres ou les espèce, voir les écomorphes. Elle caractérise alors, pour une description pré-classifiée, l'appartenance à un taxon de rang plus précis que le concept racine auquel elle est attachée.

84. Notre méthode construit des arbres et non des graphes, au contraire de l'approche proposée par Vignes (1991), qui considère des graphes d'identification : l'ensemble des noeuds terminaux qui dénotent une même classe sont unifiés, de manière à former un **graphe acyclique orienté**. Nous justifierons ce choix au § 4.6.3.

Dans ce contexte, l'objectif de la méthode d'identification est double. D'une part déterminer le concept le plus spécifique dans la hiérarchie des concepts auquel l'objet d'étude peut appartenir. Puis, dans un deuxième temps, de déterminer un nom de taxon plus précis, dans la hiérarchie de valeurs associée à l'attribut de classe du concept trouvé.

Ce double objectif répond également à une double nécessité. Celle d'offrir un outil de haut niveau pour les spécialistes du domaine, qui soit également accessible et convivial pour les non-spécialistes (techniciens de l'environnement, étudiants en biologie, amateurs, etc.). En effet, même si le but est le même, l'identification d'un échantillon, le niveau de précision recherché est différent : déterminer un nom de taxon très précis (l'espèce) pour les spécialistes, déterminer un rang plus général, la famille ou le genre, pour les autres.

Les experts savent en effet déterminer rapidement le concept de haut niveau auquel l'objet d'étude appartient, par un raisonnement synthétique global. Le méthode vient alors aider à préciser cette hypothèse initiale. Si celle-ci est erronée, l'identification conduira nécessairement à un échec ; affiner et préciser une connaissance erronée ne pouvant que conduire à un résultat moins bon encore. Cependant, se rendre compte de l'erreur initiale peut conduire à reconsidérer cette hypothèse. L'apport de l'outil informatique permet ainsi de mettre en oeuvre le processus de la découverte scientifique : « conjecturer et tester » (Popper 1973), de nature essentiellement *inductive*.

4.4.2 Principe de la méthode

La méthode d'identification (figure 4.2) est monothétique et à accès libre (Vignes 2000). Elle permet à l'utilisateur d'indiquer une à une les informations dont il dispose sur le spécimen étudié, dans l'ordre qu'il souhaite. L'arbre d'identification vient l'assister en proposant les descripteurs les plus pertinents par l'intermédiaire des noeuds de décision ; et les valeurs observables possibles, par l'intermédiaire des différentes branches. Le principe consiste donc à ajouter progressivement des éléments de description, à une description initialement vide, jusqu'à obtenir une identification.

A chaque étape, l'utilisateur dispose d'*informations contextuelles* sous la forme de photographies qui illustrent les caractères et les taxons restants. Elles permettent de mieux comprendre le domaine modélisé (glossaire, illustrations des caractères, etc.). Une fois l'identification obtenue, un ensemble d'informations concernant le spécimen est accessible, qu'elles soient internes à la base de connaissance ou bien situées dans des bases de données externes. En cas d'erreur, celles-ci permettent de reconsidérer les choix effectués, de modifier les hypothèses initiales et de *réitérer* le processus.

4.4.3 Préliminaires à une identification correcte

Les résultats fournis par le système dépendent fortement de la *robustesse* et de la *cohérence* de la BC, des choix de sélection effectués et de la qualité de la description du spécimen étudié. Les critères suivants sont les plus importants :

- **Clarté**. Le vocabulaire utilisé doit être précis, non-ambigü, expliqué.
- **Exhaustivité**. La liste des taxons présents dans la base doit être exhaustive.
- **Appartenance**. Le spécimen étudié doit appartenir à l'un des taxons décrits dans la BC.

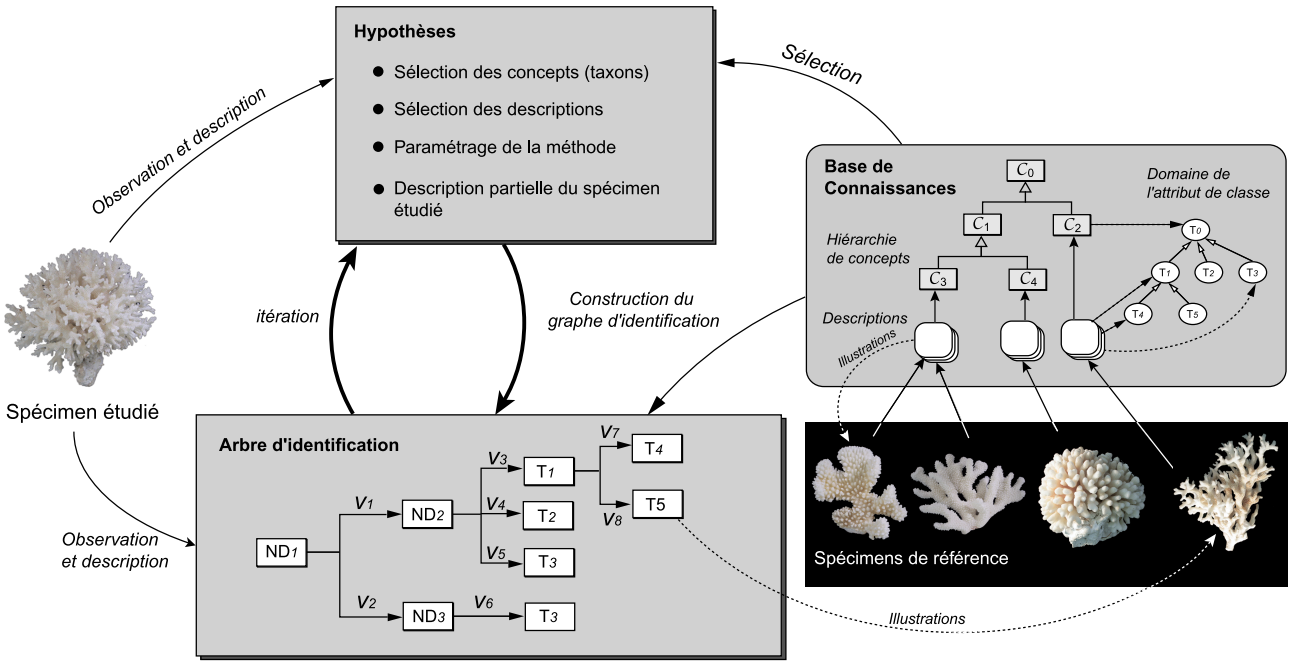


FIG. 4.2 – *Principe de l'identification avec $\mathcal{IKBS}$.*

- **Variabilité**. Les descriptions doivent couvrir la variabilité des taxons.
- **Justesse**. Les éléments de description du spécimen étudié ne doivent pas être erronés.

4.4.4 Originalité de la méthode

Les arbres d'identification sont *dynamiques* et utilisent les connaissances du domaine, pendant la phase de construction, ainsi que pendant la phase de consultation. La technologie utilisée permet de rendre accessible l'outil à tous pour effectuer des « identifications on-line ».

Construction de l'arbre d'identification

La construction de l'arbre se distingue des méthodes classiques, par les aspects suivants :

- **Relations de dépendance entre descripteurs**. Ces dépendances sont exprimées par la structure des modèles descriptifs et prises en compte lors de la construction de l'arbre d'identification.
- **Types complexes de valeurs**. Les valeurs peuvent tenir compte des réponses *inconnu*, multiples (qualitatives et quantitatives) ordonnées, classifiées ou non, imprécises, etc.
- **Identification par niveau**. Lorsque l'attribut de classe est un attribut hiérarchique, l'algorithme se fonde sur les différents niveaux de la taxonomie de valeurs, pour procéder à une identification par niveau, du plus général au plus précis.
- **Connaissances stratégiques**. Les nœuds de l'arbre sont choisis en fonction du critère classique de gain d'information, combiné avec un critère numérique attribué par le concepteur, qui reflète l'importance relative accordée à chaque descripteur. Il est également possible

de modifier l'arbre construit en sélectionnant un autre descripteur à un nœud de décision particulier. La méthode offre ainsi, dans une certaine mesure, la possibilité d'introduire des connaissances d'ordre *stratégique* dans la construction de l'arbre.

Consultation "pas à pas" de l'arbre d'identification

L'arbre d'identification n'est pas une structure autonome, isolée des descriptions et des modèles qui ont servis à sa construction. C'est également une structure d'indexation des éléments constitutifs de la base de connaissances. En effet, à chaque noeud de décision (ND) est attaché :

- la liste des descripteurs, ordonnés selon leur *pouvoir discriminant*,
- l'ensemble des descriptions qui ne sont pas encore discriminées.

Descripteurs et descriptions peuvent être consultés à chaque étape de l'identification. Chaque description étant reliée par le lien *est-un* au modèle descriptif auquel il appartient, la totalité de l'information ayant servie à la construction de l'arbre est accessible à l'utilisateur. L'utilisation de ces informations, en particulier les « connaissances de fond » offre des possibilités très intéressantes pour la phase de consultation :

- **Dynamicité**. L'arbre proposé à chaque étape est le représentant d'une forêt d'arbres [85]. Un noeud de l'arbre peut-être substitué par un autre, choisi explicitement par l'utilisateur ou déduit par le système, en cas de réponse *inconnu* ou lorsque plusieurs réponses sont possibles. L'utilisateur peut à tout moment reconsidérer ces choix et réitérer l'exploration de l'arbre.
- **Illustrations et liens hyper-textuels**. A chaque noeud de l'arbre, l'utilisateur peut accéder aux données contextuelles associées aux éléments du modèle et aux cas, afin de répondre de manière plus pertinente aux questions posés par le système. Pour les caractères macroscopiques (forme générale, aspect de surface, etc.) l'identification peut ainsi se fonder sur la photo-interprétation, à partir des photos associées aux caractères des concepts modélisés et des échantillons décrits dans la base.

Technologie utilisée

Le langage orienté-objets Java (Microsystem 2001) utilisé pour le développement de l'application $\mathcal{IKBS}$ offre une mise à disposition *en-ligne* via le Web, des arbres d'identification, d'une façon originale :

- **Accessibilité** des arbres d'identification via un navigateur Web. Java dispose de la possibilité de créer des « applications embarquées » : les **applets**. Nous avons utilisé cette technologie pour la conception de l'application autonome dédiée à l'identification. Celle-ci est téléchargée de manière transparente et exécutée sur la machine locale du client. Chaque réponse de l'utilisateur lors de la consultation est traitée rapidement par l'application, en

85. L'ensemble des arbres d'identification possible est appelé « grand graphe » par Jacques Lebbe et Régines Vignes (2000).

local. Les délais de réponse sont donc très courts, contrairement à une architecture client-serveur classique (Web dynamique), où chaque réponse est accompagnée d'une requête au serveur, suivi de l'affichage d'une nouvelle page HTML. L'outil d'identification en consultation *on-line* est donc véritablement une application côté client, dynamique et interactive.

- **Hypertexte** L'outil d'identification permet d'accéder à des ressources contextuelles internes ou externes à la base de connaissances, par l'intermédiaire d'URLs (Uniform Ressource Locator) associés aux entités de la base de connaissances. Glossaire, publications, monographies et bases de données sont accessibles par l'intermédiaire de cette indexation hypertextuelle. Par exemple, lorsque l'identification est achevée, l'utilisateur peut accéder à des bases de données externes afin d'obtenir des informations concernant le taxon trouvé : propriétés particulières de l'espèce, rôle écologique, etc.
- **Portabilité** de l'application. Java est disponible sur toute plate-forme informatique et intégré en standard dans la plupart des navigateurs Web.

4.5 Prise en compte des relations de dépendance

La prise en compte des relations de dépendances entre descripteurs permet d'introduire une logique supplémentaire, dite de bon sens, généralement absente des méthodes classiques de construction d'arbres de décision. Elle permet d'éviter certaines aberrations, lors de la consultation de l'arbre, comme par exemple de demander *quelle est la couleur des ailes ?* alors que l'individu que l'on cherche à identifier n'en possède pas. Elle résoud également un problème classique en Analyse de Données, celui de la sélection de variables liées à la présence de valeurs *inapplicables* dans l'ensemble des individus, que l'on ne sait pas traiter de manière ad hoc. En effet, ce type de valeur [86] traduit l'absence de la variable pour un individu donné, mais pas la cause de cette absence. Ce problème particulier rejoint donc une considération plus globale sur les représentations tabulaires, où seuls les individus sont considérés, indépendament de « connaissances de fond » sur le domaine considéré. Il est en effet difficile et peu naturel d'exprimer des connaissances d'ordre général sur les données, structurelles ou relationnelles, sans faire intervenir de méta-données, de connaissances supplémentaires, pour exprimer par exemple que la présence de tel descripteur (la couleur des ailes) est liée à la présence de tel autre (la présence des ailes).

4.5.1 Contexte

La prise en compte des relations de dépendance entre descripteurs dans les systèmes d'aide à l'identification a été introduite pour la première fois par (Lebbe 1991) dans l'algorithme MAKEY (cf. 4.3.3). Dans cette approche, le critère de choix d'un descripteur est relié à son « pouvoir discriminant », qui mesure le nombre de couples de concepts que ce descripteur discrimine. Cette mesure prend en compte de manière symbolique la covariance entre les descripteurs, sur laquelle les relations de dépendance influent de manière directe. Le concept de pouvoir discriminant a par ailleurs été repris par Carvalho (1991) dans le cadre de la création d'histogrammes d'objets symboliques.

86. Les **valeurs inapplicables** sont souvent notées **N-A**.

Dans le domaine de l'I.A., des extensions de l'algorithme de base d'ID_3 prenant en compte des connaissances structurées ont été développées, par exemple dans le système KATE (Manago 1991). La stratégie utilisée est cependant différente, elle consiste à ajouter une phase supplémentaire aux trois phases classiques de construction (cf. 4.3.1) pour contraindre l'espace des descripteurs pertinents à un noeud donné (Auriol 1995). Dans le même esprit, l'algorithme AlFReDO (Simon 2000) traite le problème de la sélection des attributs significatifs en se fondant sur la hiérarchie des classes à laquelle les individus sont attachés. Le principe consiste à ne prendre en compte dans un premier temps que les attributs appartenant à la *classe ascendante la plus spécifique* [87] de l'ensemble des individus à discriminer. Celle-ci ne contient que les attributs communs à toutes les sous-classes et donc à toutes les instances. Dans un second temps, un test d'instanciation peut être ajouté, permettant de tester l'appartenance des individus à certains descendants de l'APS dans le but d'augmenter les critères de sélection pouvant être sélectionnés.

4.5.2 Notre approche

Notre approche s'apparente au courant de l'I.A. pour la prise en compte des relations de dépendance entre descripteurs : une phase supplémentaire est introduite qui consiste à sélectionner la liste des descripteurs éligibles [88] pour la phase de sélection.

Dans le modèle CoDesc (chap. 2), les relations de dépendance entre descripteurs sont explicitées par les relations de composition constituant la structure d'un modèle descriptif. Un descripteur ne peut-être présent que si le composant auquel il est attaché (partie-de) est présent. La relation de composition dans ce contexte est considérée comme une relation « mère-fille », traduisant un lien existentiel entre un composant et un ensemble de descripteurs.

D'autre part, étant donné que le modèle CoDesc distingue deux types de hiérarchies auxquels sont attachées les instances : la hiérarchie des concepts $\mathcal{H}$ et la hiérarchie de classes donnée par le domaine de l'attribut de classe a_c. La question qui se pose est de savoir quelles hiérarchies utiliser pour positionner l'objet de l'étude, et de quelle manière les utiliser.

Lorsque tous les individus appartiennent au même concept, la méthode de construction revient à classer les n individus dans la hiérarchie de classes, sans tenir compte de $\mathcal{H}$. Nous discuterons au § 4.6.3 de la manière dont nous avons tenu compte de la hiérarchie induite par la structure du domaine de a_C. Les relations de dépendance entre descripteurs sont alors exprimées par le modèle decriptif dénotant le concept $\mathcal{C}$ auquel tous les individus $\{\omega_1, \ldots, \omega n\}$ que l'on cherche à discriminer appartiennent. Dans ce contexte, nous exposons deux stratégies pour la prise en compte des relations de dépendance, la seconde étant une amélioration de la première. Dans les deux cas, les relations de dépendance n'interviennent pas directement dans la mesure de discrimination.

4.5.3 Stratégie 1 : tester *a priori* la présence des descripteurs

La présence/absence des composants est considérée comme un descripteur supplémentaire, booléen, dont le pouvoir discriminant est calculé de manière classique. Ainsi, lorsque la présence

87. ou *Ascendants les plus spécifiques*, noté APS, cf.chap.3

88. **Descripteurs éligibles** : ceux qui appartiennent à la liste des descripteurs pouvant potentiellement être choisis par la mesure de discrimination.

d'un composant est confirmée, l'ensemble des descripteurs dépendants est ajouté à la liste des descripteurs pouvant potentiellement être choisis par l'algorithme.

Principe

Pour un noeud de décision N donné, l'ensemble des descriptions $\{\omega_1, \ldots, \omega_n\}$ attachées à N appartiennent tous au même concept $\mathcal{C}$. Ainsi, l'ensemble des classes cibles $C_1, \ldots, C_k$ est donné par le domaine de l'attribut de classe a_c du concept $\mathcal{C}$.

La liste L des descripteurs éligibles est construite récursivement à partir d'un composant E donné du modèle descriptif. Pour le premier noeud de l'arbre de décision, E correspond au schéma, racine du modèle descriptif. Chaque attribut nécessaire rencontré (qui appartient à un composant toujours présent) est ajouté à la liste. Lorsqu'un composant absent possible E_i est rencontré, l'algorithme de sélection s'arrête et ajoute E_i à L, qui sera interprété par la suite comme un nouveau descripteur booléen correspondant à la présence ou l'absence de E_i. Ainsi, les sous-descripteurs de E_i n'appartiennent pas à la liste L initiale et ne pourront donc pas être choisis par la procédure de sélection.

A partir d'un modèle descriptif, l'algorithme 5 procède par parcours en profondeur d'abord de ses composants suivant la relation de composition. Les conditions d'arrêts sont : lorsqu'un composant absent possible est rencontré ou lorsqu'il ne possède aucun fils (c'est une feuille), ce qui est implicite dans l'algorithme. L'algorithme utilise uniquement trois fonctions de base appliquées au composant E :

- **AbsentPossible**(E), fonction booléenne qui teste si E peut-être absent,
- **Fils**(E) qui donne la liste des fils (type Composant)
- **Attributs**(E) qui donne l'ensemble des attributs attachés à E.

Algorithme 5 Construction de la liste des descripteurs éligibles selon la stratégie 1

Données : Un composant E d'un concept $\mathcal{C}$
Résultat : La liste L des descripteurs éligibles

```
construireList1(E)
L ← Attributs(E) [L est initialisée avec les attributs directs de E]
pour tout élément {E_1,...,E_i,...,E_p} de Fils(E) faire
  si AbsentPossible(E_i) alors
    L ← E_i [Terminaison : Le composant E_i est ajouté à L]
  sinon
    L ← construireList1(E_i) [Appel récursif, les sous-descripteurs de E_i sont ajoutés L]
  fin si
fin pour
retourner L
```

A chaque noeud de décision, le nombre d'éléments de L diminue, puisque l'on retire le descripteur associé au noeud courant. Lorsque le descripteur sélectionné est de type composant E_i,

le noeud correspondant propose deux partitions correspondant à l'absence et la présence E_i. Dans la branche où la présence de E_i est avérée, un nouvel appel à **construireList1** est effectué avec E_i comme paramètre, et les éléments de la liste retournée par la fonction sont ajoutés à L. L'ensemble des descripteurs éligibles se voit alors augmenté des descripteurs qui deviennent nécessaires lorsque la présence de leur ancêtre est vérifiée.

Intérêts et limites

Cette stratégie présente l'avantage d'être relativement aisée à mettre en oeuvre et de ne jamais proposer à un noeud donné un descripteur dont la présence n'a pas été vérifiée. La complexité de la méthode est de plus linéaire, en $o(n)$, fonction du nombre n de composants parcourus, et ne dépend pas de l'ensemble des cas à discriminer. On évite ainsi le problème des valeurs *inapplicable* lors de la consultation de l'arbre de décision.

Cette stratégie présente cependant un biais d'apprentissage qui peut parfois être génant, celui de masquer un certain nombre de descripteurs à la procédure de sélection. L'efficacité de l'arbre construit [89] se trouve ainsi diminuée de la sélection possible d'un ensemble de descripteurs possédant potentiellement un fort coefficient de discrimination. En effet, ceux-ci ne se revèlent que lorsque le descripteur booléen correspondant à la présence du composant est choisi par l'étape de sélection, ce qui est le principe de base de la stratégie. Celui-ci peut posséder un pouvoir discriminant très faible ou nul, ce qui est le cas lorsque l'absence (ou la présence) est avérée pour l'ensemble des cas. Pour pallier au biais d'apprentissage de cette stratégie, nous proposons une alternative.

4.5.4 Stratégie 2 : correction du biais d'apprentissage

La stratégie 2 résoud le biais d'apprentissage exposé précédemment. L'ensemble des cas considérés forme un ensemble homogène.

Principe

Le principe consiste à considérer éligible *a priori* la totalité des descripteurs $\{a_1,\ldots,a_i,\ldots,a_p\}$ du concept, en substituant à chaque attribut contingent a_i le couple (L_{ABS},a_i) où L_{ABS} représente la liste des composants absents possibles dont la présence doit être vérifiée pour a_i. La phase de sélection se fonde alors de manière classique sur la totalité des descripteurs. Le pouvoir discriminant d'un couple est celui du deuxième élément, a_i.

Lorsqu'un couple est sélectionné, l'attribut booléen associé au premier élément E de la liste L_{ABS}, est utilisé pour le noeud de décision. Dans la branche où E est présent, le couple (L_{ABS},a_i) est à nouveau éligible, avec L_{ABS} privée de E, son premier élément. Si L_{ABS} est vide, a_i est éligible directement, la présence de tout composant *absent possible* ayant été vérifiée.

La construction de L selon ce principe peut être réalisée par l'algorithme 6 :

89. l'efficacité d'un arbre de décision peut être mesurée par la moyenne de la longueur des chemins.

Algorithme 6 Construction de la liste des descripteurs éligibles selon la stratégie 2

Données : Un composant E et une liste L_{ABS} de composants absents possible. L_{ABS} est initialement $\emptyset$.

Résultat : La liste L des descripteurs éligibles

```
construireList2(E,L_ABS)
si (L_ABS = ∅) alors
  L ← Attributs(E)
sinon
  pour tout attribut a_i de E faire
    L ← (L_ABS,a_i)
  fin pour
fin si
pour tout élément {E_1,...,E_i,...,E_p} de Fils(E) faire
  si AbsentPossible(E_i) alors
    L_ABS ← E_i
  fin si
  L ← construireList2(E_i,L_ABS)
fin pour
retourner L
```

Intérêts et limites

Cette stratégie est meilleure que la stratégie 1 dans le sens où le biais d'apprentissage exposé précédemment est évité. En effet, cette stratégie revient à substituer le pouvoir discriminant de chaque composant (traité comme un attribut booléen caractérisant la présence ou l'absence du composant) par celui du *sous-descripteur de gain maximal*, qui sera choisi par la phase de sélection. Cette stratégie peut être apparentée à la méthode adoptée par MAKEY (§ 4.3.3) pour la prise en compte des dépendances entre descripteurs.

4.5.5 Stratégie 3 : les cas sont positionnés dans une hiérarchie de concepts

Nous considérons ici la situation où l'utilisateur n'est pas en mesure de sélectionner un concept unique dans la hiérarchie des concepts.

Dans ce cas, deux sous-situations sont possibles :

- Aucune hypothèse n'est faite sur le spécimen que l'on cherche à reconnaître. C'est le cas lorsque l'utilisateur du système ne dispose d'aucune connaissance du domaine. Les stratégies précédentes ne sont alors plus applicables.
- L'utilisateur effectue quelques hypothèses, en choisissant un certain nombre de concepts et en éliminant les autres. Ce qui nécessite une connaissance du domaine. Disposer de cette connaissance permet d'orienter plus rapidement la recherche.

La figure 4.3 illustre la taxonomie de l'Ordre à la Famille des Scléractiniaires, qui correspond aux coraux durs présents dans les massifs coraliens. Un expert peut relativement facilement situer un échantillon jusqu'au niveau des familles, sans instrument particulier, les caractères macroscopiques étant généralement suffisants. La taille des calices par exemple permet de distinguer rapidement le sous-ordre des *Fungiina* des autres groupes. Ceux-ci présentent en effet des calices de plusieurs centimètres (pour les *Fungiidae*) alors qu'ils sont millimétriques pour la majorité des familles. « Descendre » jusqu'au niveau de l'espèce sans instrument adapté, tels que la loupe binoculaire est quasiment impossible [G. Faure, communication orale] pour certains groupes. Il faut de plus beaucoup de patience et de minutie.

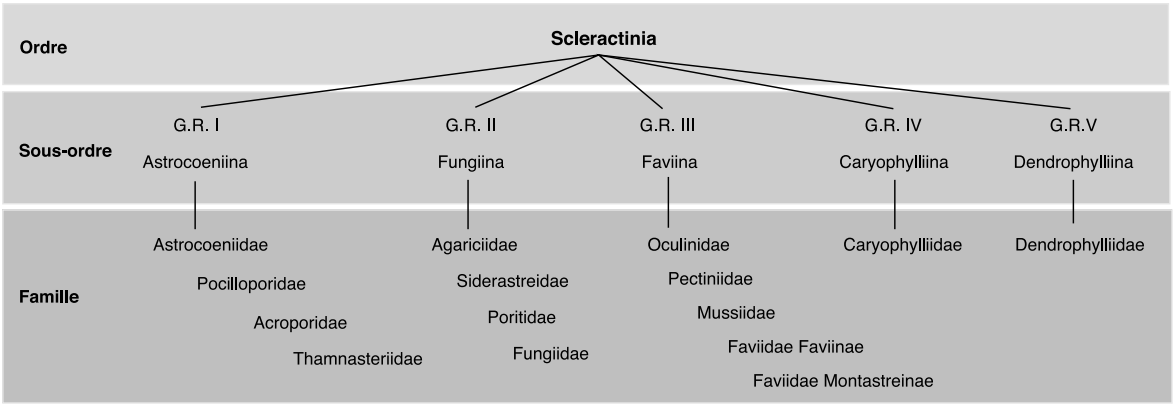


FIG. 4.3 – *Taxonomie des familles de l'Ordre des* ***Scleractinia****, d'après (Faure 1982).*

Dans la première situation, la méthode consiste à sélectionner l'ensemble d'individus *hétérogène*[90], formé par l'union de toutes les descriptions présentes dans la base de connaissances du domaine. Dans le second cas, à sélectionner l'ensemble des individus de chaque concept considéré.

Soit $\{\mathcal{C}_1, \ldots, \mathcal{C}_j, \ldots, \mathcal{C}_m\}$, l'ensemble des concepts considérés. L'ensemble d'apprentissage C est formé de l'union des individus de chaque concept considéré $C = \cup_{j<m} ext(\mathcal{C}_j)$, où $ext(\mathcal{C}_j)$ représente l'extension de $\mathcal{C}_j$, c'est-à-dire l'ensemble des individus associés $\mathcal{C}_j$.

Une fois formé l'ensemble d'apprentissage, les questions qui se posent dans ce contexte sont :

1. Sur quel ensemble de descripteurs discriminer les individus afin d'obtenir les « propriétés significatives » ?
2. lorsqu'un *sous-groupe homogène*[91] est formé à un noeud de décision, doit-on prendre en compte le sous-concept le plus spécifique ?

Recherche d'un concept généralisant

Aux chapitres 2 et 3 nous avons présenté le modèle CoDesc et les opérations de manipulation des valeurs (§ 3.3) et des concepts. En particulier, nous avons montré que pour tout ensemble de concepts, il est possible de le trouver l'Ascendant le Plus Spécifique (sec. 3.4) par l'opérateur

90. **Individus hétérogènes** : qui appartiennent à des concepts différents.
91. Par *sous-groupe homogène* nous entendons ici l'ensemble des descriptions appartenant à un concept plus précis que le concept initial.

$APS()$. Il est également possible de construire un nouveau concept, à partir d'un ensemble d'individus hétérogènes, sur la base de l'opérateur de généralisation des valeurs $GEN()$ (sec. 3.5). Dans le cas de l'APS, le concept à rechercher pré-existe et, étant donné que les hiérarchies considérées sont simples, il est relativement aisé et rapide de le trouver. Dans le cas du concept généralisé à partir d'individus, l'opération est plus complexe et plus coûteuse (cf. chapitre 3), mais présente l'avantage d'être plus précis que l'APS dans le cas ou l'ensemble est « très hétérogène ».

Une fois le concept trouvé ou formé, quelle que soit la méthode utilisée, les propriétés significatives sont les descripteurs du concept. Une réponse à la première interrogation est ainsi obtenue. Les stratégies 1 et 2 exposées précédemment sont alors applicables. Un seul concept est donc considéré, qu'il soit déterminé par l'utilisateur ou bien calculé par le système.

Prise en compte des concepts plus spécifiques

A l'inverse des représentations à objets classiques, en particulier de la méthode AlFreDo exposée précédemment, il n'est pas nécessaire de chercher à détecter si les individus restant à discriminer à un noeud donné forment un groupe homogène d'individus. Les Descendants les Plus Généraux (DPG) ne sont donc pas considérés.

Etant donnée la définition de la spécialisation des concepts de CoDesc (§ 2.6), la hiérarchie de concepts est construite par union des descripteurs (éventuellement contingents) de tous les sous-concepts. Ainsi, tout descripteur de tout individu peut être pris en compte par la phase de sélection, quel que soit le concept auquel l'individu appartient. La donnée d'un unique concept est donc suffisante pour conduire à une identification pertinente, ce qui répond à la seconde interrogation.

4.5.6 Discussion

Dans une approche « purement objet », il est nécessaire de détecter des groupes homogènes d'individus, afin de conduire une identification pertinente (Simon 2000, Simon & Napoli 1997). En effet, les classes des modèles objets sont définies en intension, par l'ensemble des descripteurs communs à toutes les sous-classes. Omettre de détecter des groupes d'individus homogènes (par un test d'instanciation) lorsque l'ensemble d'apprentissage initial est très hétérogène, reviendrait à discriminer les individus sur la base d'un nombre restreint et pas nécessairement discriminant d'attributs : ceux qui sont présents dans les classes, constituent les ancêtres communs les plus spécifiques. Comparativement, notre approche présente l'intérêt de ne pas faire intervenir de *test d'instanciation*[92] lors de la phase de sélection des attributs éligibles, et donc de se fonder uniquement sur les propriétés d'un concept, sans faire intervenir les individus. Ceux-ci n'interviennent que pour déterminer le concept commun, ce qui est un préliminaire à la construction de l'arbre.

Les stratégies de sélection des descripteurs éligibles présentées sont donc efficaces et de faible complexité, puisqu'elles ne dépendent aucunement des individus à discriminer ; uniquement de la donnée d'un seul concept dont l'extension recouvre l'ensemble d'apprentissage. Celui-ci peut être déterminé par le système. Ce point est important en terme de complexité algorithmique

92. Test d'instanciation : opérateur booléen permettant de tester l'appartenance d'une description à une classe donnée.

puisque la sélection des descripteurs éligibles est effectuée pour la construction de chaque noeud de décision.

4.6 Complexité des valeurs

Comme nous l'avons montré au chapitre 2, l'état d'une propriété pour un attribut donné peut être simple, de type ensemble, une conjonction, non renseigné, *inconnu* ou encore exceptionnel (c'est-à-dire non défini dans le domaine de l'attribut). Un attribut peut être de type symbolique, hiérarchisé, numérique, ou encore définir un champ de saisie de caractères (type textuel).

Cette diversité des types d'attributs et de valeurs pose quelques difficultés pour la construction de l'arbre, en particulier :

- Pour le choix des valeurs constitutives de la partition. Celles-ci forment les étiquettes des arcs orientés entre deux noeuds de décision.
- Pour le partitionnement des descriptions en n sous-ensembles.

Dans l'approche classique, l'ensemble des cas (descriptions) attachés à un noeud de décision donné est découpé en n sous-ensembles disjoints, correspondant aux éléments de $\mathbf{dom}(A)$, A étant le descripteur sélectionné, le plus discriminant au noeud de décision.

4.6.1 Construction d'une partition

Soit ND un noeud de décision. Sont attachés à ce noeud :

- Un ensemble de descriptions $P = \{\omega_1, \ldots, \omega_i, \ldots, \omega_n\} \subseteq \Omega$.
- La liste des attributs éligibles $L = \{A_1, \ldots, A_j, \ldots, A_p\}$, ordonnée par ordre décroissant, selon le gain d'information[93] respectif de chaque attribut. A_1 est ici le descripteur le plus discriminant au noeud ND.

La construction des différentes branches se fonde sur l'ensemble des valeurs simples associées à l'attribut sélectionné. Nous considérons que les valeurs simples (ou atomiques) sont les éléments de $\mathbf{dom}(A)$, c'est-à-dire les valeurs de premier niveau du treillis de valeurs associées au co-domaine de A (cf.§ 2.8), noté $\mathbf{cod}(A)$ et ceci quelque soit le type de A : numérique, nominal, ordinal ou hiérarchique. Ainsi, si $\mathbf{dom}(A) = \{bleu, jaune, rouge\}$, les valeurs simples associées à A sont *bleu*, *jaune* ou *rouge*.

Descripteurs nominaux et ordinaux

Admettons que A_1 soit de type nominal ou ordinal. L'ordre des valeurs n'étant pas pris en compte dans la méthode (les partitions ne sont pas ordonnées), un descripteur ordinal est traité de la même manière qu'un nominal.

Chaque valeur simple $v_i^{A_1}$ de $\mathbf{dom}(A_1) = \{v_1^{A_1}, \ldots, v_j^{A_1}, \ldots, v_k^{A_1}\}$ défini k-branches représentées chacune par un arc orienté partant de ND qui est étiqueté par v^{A_1}. Les valeurs non

93. La mesure de gain d'information que nous avons utilisée est le critère de l'**entropie de Shannon**.

exprimées dans l'ensemble P sont élaguées de l'arbre *a posteriori* et ne forment donc pas de branche. Plus formellement, les valeurs retenues pour constituer une partition sont l'ensemble des $v_j^{A_1} \in \mathbf{dom}(A_1)$, telles que :

$$\exists v_{\omega_i}^{A_1},\ i = 1, \ldots, n,\ /\ v_j^{A_1} \in v_{\omega_i}^{A_1}$$

Nous rappelons que toute valeur v_{ω_i} est considérée comme un ensemble, étant donnée la définition des attributs, vus comme des applications à valeur dans l'ensemble des parties de leur domaine (cf.§ 2.8), ce qui justifie le test $v_k^{A_1} \in v_{\omega_i}^{A_1}$ et non pas le test d'égalité utilisé par les algorithmes classiques.

Après sélection, le descripteur A_1 est éliminé de la liste des attributs éligibles.

Descripteurs numériques

Si A_1 est de type numérique, l'intervalle de définition de A est $\mathbf{dom}(A) = [binf \neq\ bsup]$. La méthode consiste simplement à diviser $\mathbf{dom}(A)$ en deux intervalles disjoints, $[binf,\ c]$ et $]c,\ bsup]$ qui permettent d'obtenir la meilleure partition de P. Nous ne développons pas la technique qui est bien connue en apprentissage supervisé (Quinlan 1992).

Notons qu'après sélection, A_1 n'est pas nécessairement éliminé de la liste des descripteurs éligibles. Cela dépend de la réponse de l'utilisateur. Celui-ci dispose en effet de deux alternatives :

- Répondre une valeur précise observée sur le spécimen étudié. A_1 est alors éliminé et la valeur renseignée est mémorisée dans la description courante du spécimen.
- Choisir une des deux partitions proposées : $[binf,\ c]$ et $]c,\ bsup]$. A_1 n'est pas éliminé, mais son domaine de valeur est restreint à l'une des deux partitions. Ainsi, il peut être à nouveau choisi par la fonction de sélection dans le but d'affiner la description initiale, par dichotomies successives.

Descripteur hiérarchique

Pour les descripteurs de type hiérarchique, la technique utilisée pour la construction des différentes partitions est proche de celle utilisée pour les attributs nominaux. Deux stratégies sont possibles selon que l'ordre partiel de la hiérarchie de valeurs est pris en compte ou non. Ce choix est défini par l'utilisateur lors du paramétrage de la méthode, avant construction de l'arbre d'identification.

- Le domaine de A_1 est « aplati ». Chaque noeud de la taxonomie associée à A_1 est considéré comme une valeur indépendante, la relation d'ordre partielle n'est donc pas prise en compte. A_1 est alors traité comme un descripteur nominal puis éliminé après sélection.
- Le domaine associé à A_1 est décomposé en plusieurs niveaux, du plus général au plus particulier. Lorsque A_1 est sélectionné pour la première fois, les différentes branches sont constituées des valeurs de premier niveau de la taxonomie. Pour chacune des branches, une copie de A_1 est insérée dans la liste des descripteurs éligibles. Pour chaque copie, une restriction du domaine est effectuée, correspondant aux valeurs de second niveau les plus spécifiques de la valeur utilisée pour définir la branche courante.

Dans la seconde stratégie, le principe est comparable au traitement des descripteurs numériques, la valeur est précisée successivement, du plus général au plus précis. L'utilisateur peut également renseigner une valeur précise observée du spécimen d'étude qui est alors ajoutée à la description globale.

Nous détaillons ci-dessous (§ 4.6.3) le principe de l'identification par niveau lorsque l'attribut de classe (variable cible) choisi est hiérarchique. La philosophie est similaire et peut aider à mieux comprendre le principe exposé quelque peu brièvement ici.

Prise en compte des valeurs conjonctives

Le cas des valeurs conjonctives pose un problème d'interprétation délicat à résoudre dans le contexte de la construction d'arbres d'identification. Ce problème est évoqué dans (Auriol 1995), nous l'avons souligné au § 2.8.

Etant donné une valeur conjonctive, par exemple $v = bleu \wedge jaune$, la sémantique que nous lui accordons est la présence simultanée des deux états *bleu* et *jaune* (variation), pour une même observation. Ce qui est différent de la disjonction, notée $v = bleu \vee jaune$ que nous assimilons à $v = \{bleu,jaune\}$ et qui traduit une *imprécision*, c'est-à-dire l'incapacité de l'observateur à déterminer clairement entre les deux couleurs. Nous rappelons ici le principe général qui est :

« Une conjonction de valeurs exprime une variation, une disjonction exprime une imprécision »

La sémantique étant donnée, le problème qui se pose pour la construction des branches de l'arbre est celui de l'interprétation que nous avons évoqué au § 2.8. L'utilisateur effectue le choix de la manière dont seront traitées les valeurs conjonctives, avant la construction de l'arbre :

- Une conjonction est une nouvelle valeur, indépendante des valeurs simples qui la composent. Toute valeur conjonctive présente dans les observations forme une nouvelle branche.
- Une conjonction est une valeur plus spécialisée que les valeurs simples. Le domaine de l'attribut forme implicitement une taxonomie de valeurs.
- Une conjonction est traitée de la même manière qu'une disjonction.

Dans la première acception, le nombre de partitions augmente rapidement, ce qui complexifie fortement l'arbre d'identification et a pour effet de diminuer le pouvoir prédictif de l'arbre d'identification et de détériorer la qualité des résultats d'identification. La seconde interprétation nous semble la plus satisfaisante. Elle est adoptée par défaut pour la construction de l'arbre d'identification. Enfin, la dernière interprétation donne également de bons résultats. Nous illustrerons ces résultats par une expérimentation au § 4.9.

4.6.2 Affectation des descriptions

Comme nous l'avons vu au § 2.8, toute valeur peut-être interprétée comme un élément du treillis de valeurs associé au co-domaine d'un attribut. La relation d'ordre est interprétée comme une relation de généralisation / spécialisation ; le plus petit élément du treillis est la valeur *inconnu* et le plus grand élément est l'ensemble vide. Cette structure nous donne un moyen pratique pour déterminer l'affectation des descriptions à chaque partition :

Dans le but de constituer une partition en k sous-ensembles de P, l'affectation des descriptions à chaque sous-ensemble se fonde sur l'opérateur de généralisation des valeurs, ce qui a pour effet de plonger les valeurs complexes dans un cadre commun : la relation d'ordre partielle induite par la relation de généralisation.

Etant donné un noeud de décision ND,

A_1 le descripteur le plus disciminant,

$P = \{\omega_1, \ldots, \omega_i, \ldots, \omega_n\}$, l'ensemble des descriptions,

$I = \{v_1^{A_1}, \ldots, v_j^{A_1}, \ldots, v_k^{A_1}\}$ l'ensemble des k valeurs de partitions

et $\{P_1, \ldots, P_j, \ldots, P_k\}$ k-sous-ensembles correspondant aux valeurs de I.

L'affectation des descriptions aux k-sous-ensembles est telle qu'il existe une chaîne dans le treillis reliant $v_{\omega_i}^{A_1}$ et $v_j^{A_1}$. Cette condition peut s'exprimer sous la forme de deux sous-conditions, l'une ou l'autre devant être respectée :

Définition 4.1 *Condition d'affectation, $v_j^{A_1}$ est plus spécifique que $v_{\omega_i}^{A_1}$.*

$$\forall v_{\omega_i}^{A_1} \in P \ , \ \forall v_j^{A_1} \in I \ , \ v_j^{A_1} \leqslant v_{\omega_i}^{A_1} \Rightarrow \omega_i \in P_j$$

$v_{\omega_i}^{A_1}$ est l'élément maximal de la chaîne.

Définition 4.2 *Condition d'affectation, $v_j^{A_1}$ est plus générale que $v_{\omega_i}^{A_1}$.*

$$\forall v_{\omega_i}^{A_1} \in P \ , \ \forall v_j^{A_1} \in I \ , \ v_{\omega_i}^{A_1} \leqslant v_j^{A_1} \Rightarrow \omega_i \in P_j$$

$v_j^{A_1}$ est l'élément minimal de la chaîne.

Discussion

Quelques remarques sur ces définitions :

- La partition ainsi obtenue n'est pas nécessairement disjointe. Une description ω peut être affectée à plusieurs sous-ensembles P_j de P, et donc participer à la construction de plusieurs sous-arbres.
- Une description pour laquelle la valeur de A_1 est *inconnu* (?) ou *non renseigné* ($\emptyset$) est affectée à chaque sous-ensemble. En effet, pour toute valeur du treillis, il existe toujours au moins un chemin avec la valeur **?** (élément minimal du treillis) et avec $\emptyset$ (élément maximal du treillis). La valeur *non renseigné* est donc traitée comme la valeur *inconnu*. Le paramétrage de la méthode offre la possibilité d'éliminer les descriptions pour lesquelles la valeur de A_1 est $\emptyset$, pendant le traitement.
- La classe cible à laquelle appartient une description peut être atteinte en suivant plusieurs chemins de l'arbre.
- La première définition s'applique pour les valeurs de type ensemble, plus générales que les valeurs simples des différentes branches.
- La seconde définition s'applique pour les valeurs hiérarchiques des descriptions, plus précises (donc plus spécifiques) que les valeurs les plus générales utilisées pour définir les branches.

4.6.3 Identification par niveau

Considérons un ensemble homogène d'individus, pour lequel l'attribut cible (ou attribut de classe) est de type hiérarchique. C'est le cas lorsque l'on cherche à déterminer le nom d'un individu parmi un ensemble de noms taxonomiques, organisés hiérarchiquement. L'attribut de classe du modèle (la cible) est alors appelé *taxon*, et son domaine de valeur caractérise une hiérarchie locale : la valeur la plus générale correspondant au concept auquel appartiennent les individus, les valeurs les plus précises, aux espèces ou aux écomorphes. La figure 4.4 illustre un exemple d'attribut de classe hiérarchique : la taxonomie de la famille des *Pocilloporidae* (coraux branchus).

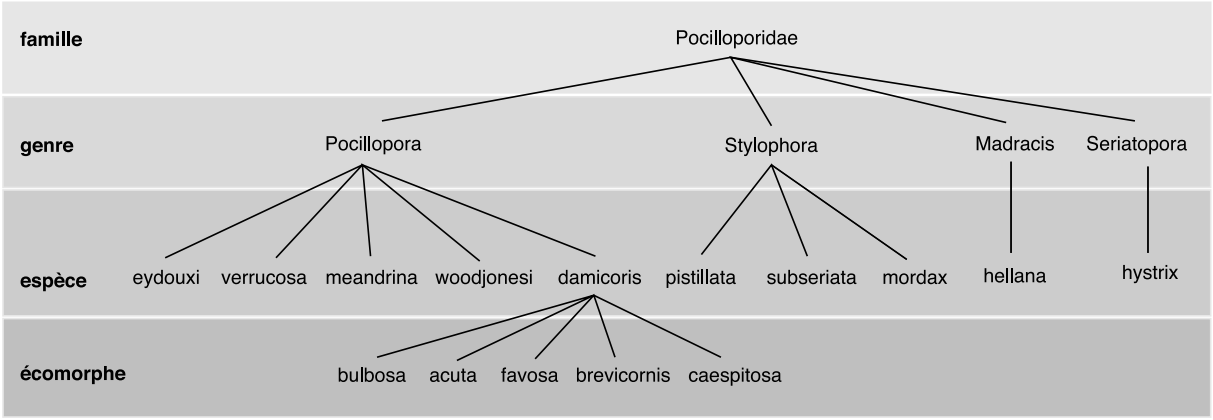


FIG. 4.4 – *Taxonomie de la famille des Pocilloporidae, d'après (Faure 1982).*

Dans la plupart des méthodes d'identification par arbre de décision, l'attribut cible est de type nominal et aucune relation d'ordre n'existe entre les valeurs. Dans l'exemple 4.4, la solution consiste à « aplatir le domaine », ce qui revient à considérer que :

$$\mathbf{dom}(a_c) = \{Pocillopora, Stylophora, Madracis, Seriatopora, eydouxi, verrucosa, etc.\}$$

Aucune relation n'est donc pris en compte entre les valeurs. Par exemple entre *eydouxi*, qui est une espèce appartenant au genre *Pocillopora*.

Dans la pratique, il est préférable pendant la phase de consultation, de procéder par étapes successives, c'est-à-dire de déterminer dans un premier temps une classe assez générale (par exemple la famille ou le genre), puis progressivement affiner l'identification pour arriver jusqu'au niveau de précision recherché.

La solution que nous proposons permet d'effectuer une identification par étapes successives en introduisant un type de noeud de décision supplémentaire, appelé *feuille intermédiaire*.

Feuille intermédiaire

Une feuille intermédiaire est simultanément :

- une feuille qui correspond à une *classe intermédiaire* de la hiérarchie,
- un noeud de décision. Lorsqu'elle est activée, un sous-arbre est développé. Les feuilles du sous-arbre sont des classes intermédiaires ou des classes terminales (feuilles de la hiérarchie de valeurs).

La figure 4.5 illustre le développement d'une feuille intermédiaire pendant la phase de consultation. Les genres *Stylophora* et *Pocillopora* sont deux feuilles intermédiaires. La feuille *Pocillopora* est développée pour conduire l'identification jusqu'au niveau des espèces *Pocillopora verrucosa* et *Pocillopora meandrina*.

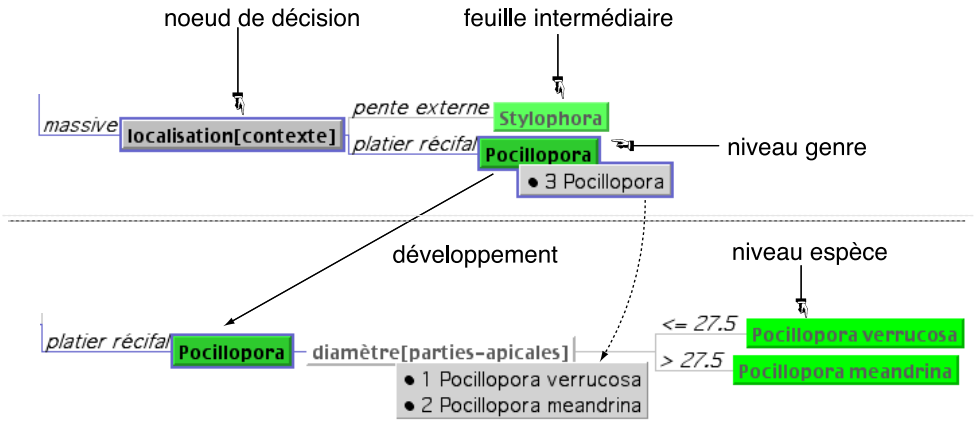


FIG. 4.5 – *Développement d'une feuille intermédiaire pendant la phase de consultation.*

Méthode

La méthode est relativement simple. Elle consiste à créer un attribut de classe temporaire, de type nominal, à partir de l'attribut de classe taxonomique initial. Le domaine de cet attribut intermédiaire est constitué des valeurs de premier niveau de la hiérarchie de valeurs.

Lors du développement d'une feuille intermédiaire, le sous-arbre de la hiérarchie de valeurs dont la racine est la valeur associée à la feuille intermédiaire est considéré. Le principe de construction de l'arbre est à nouveau appliqué à partir de ce sous-arbre, récursivement.

Exemple

Considérons l'attribut de classe $Taxon$ illustré à la figure 4.4.

Premier niveau : identification du genre. Un attribut de classe temporaire :
$Pocilloporidae \rightarrow \mathbf{dom}(a_c^1) = \{Pocillopora, Stylophora, Madracis, Seriatopora\}$

Second niveau : identification de l'espèce. Quatre attributs de classe temporaires sont créés :
$Pocillopora \rightarrow \mathbf{dom}(a^{21})_c = \{eydouxi, verrucosa, meandrina, woodjonesi, damicornis\}$
$Stylophora \rightarrow \mathbf{dom}(a^{22})_c = \{pistillata, subseriata, mordax\}$

$Madracis \rightarrow \mathbf{dom}(a^{23})_c = \{hellana\}$
$Seriatopora \rightarrow \mathbf{dom}(a^{24})_c = \{hystrix\}$

Troisième niveau : identification des écomorphes. Un attribut de classe temporaire :
$damicornis \rightarrow \mathbf{dom}(a^{3})_c = \{bulbosa,acuta,favosa,brevicornis,caespitosa\}$

Discussion

Cette technique est relativement simple à mettre en oeuvre au sein d'un méthode d'identification monothétique. Elle est fortement appréciée par les biologistes car elle correspond à une démarche logique du processus d'identification. Elle introduit des résultats intermédiaires qui peuvent être infirmés ce qui simplifie le processus.

L'introduction de ce type de noeud dans l'arbre d'identification justifie le choix de ne pas considérer des graphes acycliques orientés (en unifiant les feuilles qui dénotent des classes identiques), comme dans (Vignes 1991, Vignes 2000), mais bien des arbres. En effet, les feuilles intermédiaires dénotant une même classe ne peuvent être unifiées du fait que la liste des descripteurs éligibles et la liste des cas non discriminés à ce stade sont différentes.

4.7 Connaissances d'ordre stratégique

Les critères optimisés lors de la construction de clés par un système d'I.A.O. ou manuellement ne sont pas toujours les mêmes. Par exemple, après avoir proposé quelques caractères faciles à observer, une clé manuelle élimine souvent rapidement les taxons qui ont une morphologie originale, puis essaie de discriminer les taxons restants, en respectant une classification particulière que l'auteur de la clé veut illustrer. Dans les systèmes d'I.A.O., le critère utilisé à le mérite d'être clair (gain d'information), mais reste identique à chaque étape de la construction de la clé. Le systématicien n'a généralement pas la possibilité de modifier la clé générée, en fonction de l'idée qu'il a de la clé qu'il souhaite obtenir (Vignes 2000).

Dans l'approche que nous proposons, la construction de la clé est fondée essentiellement sur les descriptions de spécimens. La clé générée dépend alors très fortement de l'ensemble d'apprentissage sélectionné et de la représentativité des différents concepts présents dans les données. Dans ce contexte, il est important de donner la possibilité d'orienter le descripteur sélectionné par le critère de choix, avant la construction de l'arbre. Ceci est rendu possible de plusieurs manières :

- En éliminant temporairement certains descripteurs, avant la construction de l'arbre.
- En associant des poids aux descripteurs du modèle descriptif (concept) sélectionné, avant la construction de l'arbre.
- Après construction, le descripteur sélectionné à un noeud de décision donné peut être modifié, c'est-à-dire choisi explicitement par l'expert.

4.7.1 Elimination ou modification du poids des descripteurs

L'expert peut juger que certains attributs sont peu signifiants, bruités ou sources d'erreurs. En effet, la conception d'un modèle descriptif n'est pas uniquement fondée sur des considérations liées à la génération de clés d'identification. Pour les coraux par exemple, l'expert souhaite que les descriptions renseignent également les aspects suivants :

- Gestion de collection. Le numéro de l'échantillon, le nom de la personne qui a effectué la description, etc.. Ces descripteurs sont clairement non signifiants pour la discrimination.
- Aspects contextuels. L'état de la colonie (bon état, mauvais état), la profondeur de récolte, le lieu de récolte etc.. Ces caractères sont généralement non signifiants, mais dans certains cas, l'expert considère qu'ils font partie intégrante de la description.

L'expert élimine donc ces descripteurs *a priori*. Dans le même ordre d'idée, il peut adopter un point de vue moins radical qui consiste à diminuer ou au contraire augmenter l'importance de certains descripteurs, en introduisant des poids d'importance relative.

Soit $p_A \in \mathbb{R}$, le poids associé au descripteur $A \in \mathcal{A}$. Nous avons fixé $0 \leq p_A \leq 2$. 0 correspond à l'élimination du descripteur, 1 au poids par défaut associé à tout descripteur et 2 au poids maximal pouvant être associé.

Le poids associé à un descripteur est ensuite combiné avec le critère de gain d'information (quel que soit ce critère)[94], pour obtenir le *gain pondéré* ($gainP(A)$) qui sera utilisé lors de la phase de sélection des descripteurs.

Définition 4.3 ***Gain pondéré***

Soit un noeud de décision donné ND, $A \in \mathcal{A}$ l'attribut le plus discriminant selon la mesure du gain d'information $gain(A)$ (toujours positif), et $p_A \in [0\ 2]$ le poids associé à A.

Le gain pondéré est calculé simplement comme le produit du gain d'information par le poids associé à A :

$$gainP(A) = p_A * gain(A)$$

Définition 4.4 ***Gain ratio pondéré***

Soit $I(A)$, l'intervalle des valeurs possibles de $\mathbf{dom}(A)$. Pour A de type discret (symbolique, hiérarchique, etc.), $I(A)$ correspond au cardinal de $\mathbf{dom}(A)$; pour A continu, $I(A)$ correspond au nombre d'éléments obtenu par discrétisation de $\mathbf{dom}(A)$. Se référer à Quinlan (1996) pour les détails concernant la manière de définir n seuils à partir de l'intervalle $\mathbf{dom}(A)$ d'un attribut continu.

Le gain ratio pondéré (choix par défaut dans notre méthode) est donné par :

$$gainRatioP(A) = \frac{p_A * gain(A)}{I(A)}$$

94. Nous utilisons par défaut le *gain ratio* ou *rapport de gain*, qui est une normalisation du critère de l'*entropie* de Shannon (détaillé dans Khinchin (1957)).

Associer des poids aux descripteurs permet ainsi de qualifier l'importance relative des uns par rapport aux autres et ainsi d'introduire des connaissances stratégiques dans la construction des clés afin en guider la construction de l'arbre d'identification.

4.7.2 Approche semi-automatique

L'expert peut également choisir, après construction de l'arbre, le descripteur qui lui semble le plus pertinent, à un noeud de décision donné ND_1. Le sous-arbre ND_1 de racine est alors substitué par un autre noeud de décision ND_2 correspondant au descripteur choisi. En appliquant méthodiquement cette action de la racine de l'arbre initial jusqu'au noeuds de décision terminaux, l'expert peut construire manuellement l'arbre. L'approche est alors semi-automatique, c'est-à-dire qu'une fois construit automatiquement, l'arbre peut être modifié manuellement en partie ou dans sa totalité.

Notons que certains auteurs proposent des programmes ayant le même principe, permettant à l'expert de valider ou modifier chaque noeud proposé au fur et à mesure de la création de la clé (Pankhurst 1998).

4.7.3 Dynamicité de l'arbre

L'arbre proposé à chaque étape est le représentant d'une forêt d'arbres. Un noeud de l'arbre peut être substitué par un autre, choisi explicitement par l'utilisateur ou déduit par le système, en cas de réponse *inconnu* ou lorsque plusieurs réponses sont possibles. L'utilisateur peut à tout moment reconsidérer ses choix et réitérer l'exploration de l'arbre. Le sous-arbre local est reconstruit en conséquence, dynamiquement. Ainsi l'exploration de l'arbre d'identification revient à explorer le « grand graphe »(Causse & Lebbe 1995), défini par Vignes (2000) comme :

Le « grand graphe » est déterminé de manière unique pour chaque base de connaissances : il représente tous les graphes d'identification potentiellement constructibles. Ce graphe contient deux types de noeuds, les « noeuds-descripteurs » qui correspondent à chaque contexte d'identification possible. De ces noeuds partent autant d'arcs que de descripteurs pertinents (ou éligibles) dans ce contexte. Les « noeuds-valeurs » [95] représentent toutes les valeurs que peut prendre un descripteur dans le contexte local d'identification.

Ainsi l'arbre d'identification proposé masque une forêt d'arbres possibles (le grand graphe ou grand arbre dans notre contexte). Chaque noeud de décision contient la liste des descripteurs pertinents, classés selon un critère de choix (gain d'information, GINI index, etc.). Ce critère établit un préordre sur l'ensemble des descripteurs.

4.8 Consultation « en-ligne » de l'arbre d'identification

4.8.1 La technologie des *applets Java*

La technologie Java utilisé pour la conception du système $\mathcal{IKBS}$ permet la mise à disposition « en-ligne »de la méthode d'identification. Tout utilisateur dispose ainsi de la possibilité

95. Les **noeuds-valeurs** correspondent aux noeuds de décision (ND) dans notre contexte.

d'identifier des spécimens à l'aide d'un simple accès au Web, ce qui augmente de manière conséquente l'*accessibilité* aux produits de nos recherche. Java dispose de la possibilité de créer des « applications embarquées » : les **applets**. Ces applications sont téléchargées de manière transparente et exécutée sur la machine locale du client. Chaque réponse de l'utilisateur lors de la consultation est traitée rapidement, car localement. Les délais de réponse sont donc très courts, contrairement à une architecture client-serveur classique (Web dynamique), où chaque réponse est accompagnée d'une requête au serveur, suivi de l'affichage d'une nouvelle page HTML.

Ainsi « embarquée » la méthode d'identification conserve également toute ses fonctionalités : dynamicité de l'arbre, en cas de réponse *inconnu*, accès aux descriptions et aux modèles descriptifs, etc..

L'outil d'identification en consultation *on-line* est donc véritablement une application côté client, dynamique et interactive (cf. fig. 4.6).

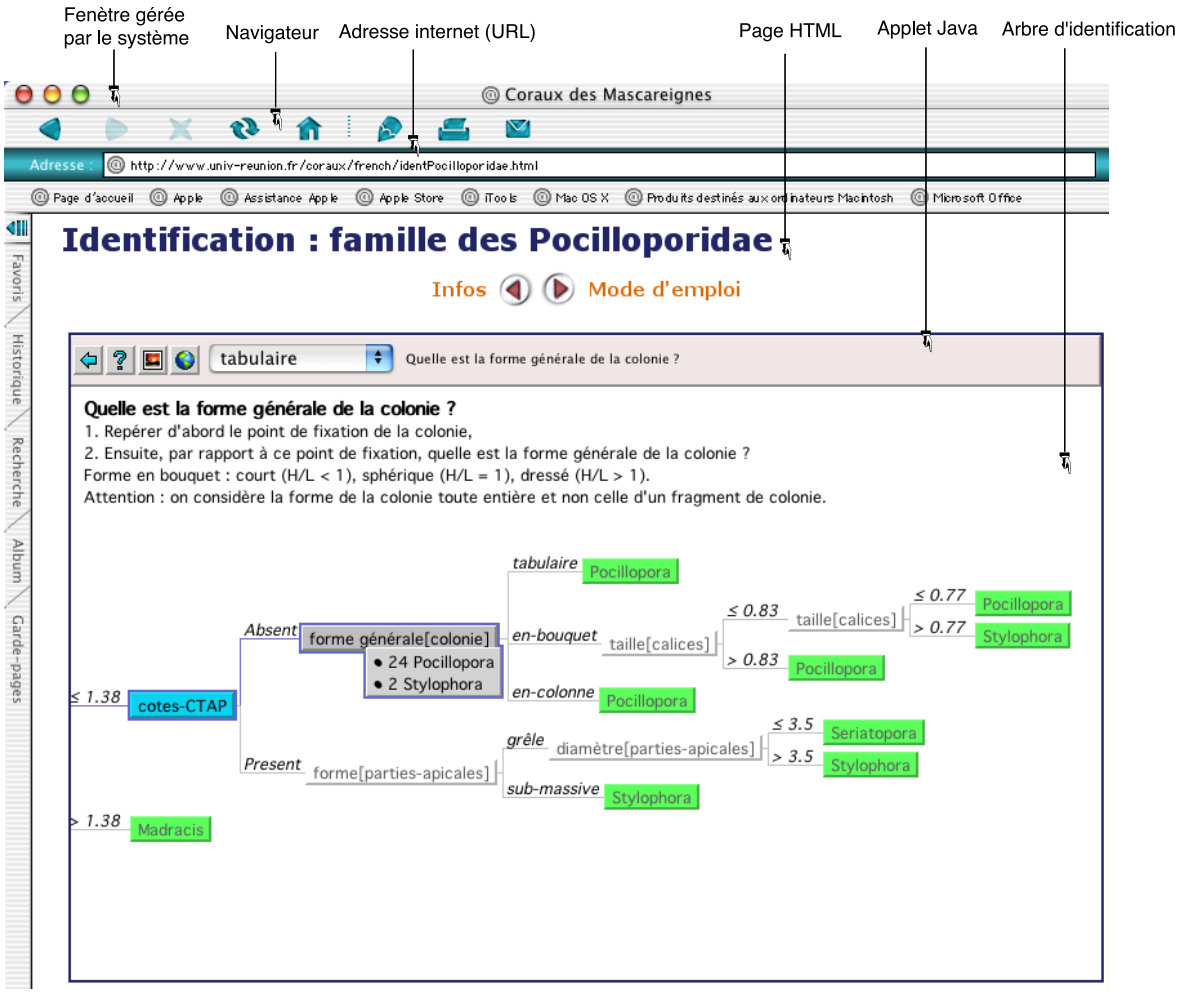


FIG. 4.6 – *Identification en-ligne de l'arbre d'identification*

4.8.2 Accès aux données contextuelles

L'outil d'identification permet d'accéder à des ressources contextuelles internes ou externes à la base de connaissances, par l'intermédiaire des liens de référence définis dans la base de connaissances. Dans ce contexte « orienté-web », ces références sont de type URL (Uniform Ressource Locator). Glossaire, publications, monographies et bases de données sont accessibles par l'intermédiaire de cette indexation hypertextuelle. Par exemple, lorsque l'identification est achevée, l'utilisateur peut accéder à des bases de données externes afin d'obtenir des informations concernant le taxon trouvé : propriétés particulières de l'espèce, rôle écologique, etc.

L'annexe B présente quelques exemples d'illustration des caractères de la famille des *Pocilloporidae* au format HTML.

4.9 Expérimentation

4.9.1 Objectifs

Nous proposons une étude visant à :

- d'une part estimer la fiabilité de la méthode d'identification monothétique proposée, sur différents jeux de données,
- d'autre part, à illustrer de manière empirique le gain du pourcentage de cas bien classé, lorsque la structuration des descripteurs est prise en compte ou non.

Les données

Les données sont issues d'une des bases de connaissances modélisées avec $\mathcal{IKBS}$ et d'autre part du l'*UCI Machine Learning Repository*(Merz & Murphy 1996). Ce sont des bases standards mises à disposition et téléchargeables pour tester des méthodes d'apprentissages automatique ou d'Analyse de Données. Nous avons sélectionné 17 de ces bases, afin d'illustrer le comportement de la méthode d'identification sur des bases hétérogènes : nombre de cas variable (entre 26 et 10000), et types de descripteurs hétérogènes (quantitatif, nominal, ordinal). Tout individu de chaque base est préclassifié ; une variable parmi l'ensemble des descripteur est la **classe** à déterminer (*attribut de classe* ou *conclusion*). C'est la variable que l'on cherche à prédire.

4.9.2 Principe

Le principe consiste à effectuer une validation croisée automatique, en suivant les étapes suivantes :

Pour une base donnée, à partir du premier cas :

- Choisir un cas,
- Construire un arbre de décision (en interne, c'est-à-dire sans interface graphique),
- Identifier le cas choisi à l'aide de l'arbre construit, en aveugle, c'est-à-dire que la classe associée au cas choisi n'est pas considérée,

– Comptabiliser l'erreur du résultat d'identification,
– Réitérer le processus sur l'ensemble des cas de la base et calculer le pourcentage de cas bien classés au final.

4.9.3 Expérience 1 : sans utilisation des structures

Le tableau suivant (4.1) illustre les résultats d'identification sur les différentes bases. La structuration n'est pas prise en compte, c'est-à-dire que les descripteurs hiérarchiques sont "transformés" en nominaux, et les relations de dépendance non prises en compte. Toutes les bases sont des tableaux de données.

Bases	Cas	Quantitatif	Nominal	Ordinal	**Résultats**
Annealing	798	9	29		95,73%
Audiology	200		69		72,80%
Audiology test	26		69		87,12%
Bridges	108	4	5	2	54,60%
* *Pocilloporidae* (coraux)	113	30	112	15	**61,64**%
* *Siderastreidae* (coraux)	60	24	61	10	**81,83**%
* *Fungiidae* (coraux)	63	4	39		**78,00**%
Echocardiogram	132	7	2	0	79,78%
Flag	194	10	18	0	43,15%
Hepatitis	155	6	13	0	76,50%
Images segmentation	420	18	1	0	97,04%
LED+17 noise	10000	0	24	0	63,91%
Monks-1	432		6		81,59%
Monk2-2	432		6		96,30%
Monk2-3	432		6		100,00%
Mushroom	8124	1	21		100,00%
Soybean (large)	307	6	29		87,56%
Soybean (small)	47	6	29		100,00%
* *Hyalonema* (éponges)	60	24	41	10	**56,80**%
Vehicle	846	18			77,60%
Zoo	90		16		97,67%
Moyenne					70,40%

TAB. 4.1 – *Résultat d'identification monothétique par validation croisée. * sont des bases du projet BC Coraux. Les autres sont issues de l'UCI Machine Learning Repository.*

Interprétation

Le lecteur pourra comparer ces résultats avec d'autres méthodes de détermination [96] accessibles sur le site de l'*UCI Machine Learning Repository.*

96. par exemple les méthodes de discrimination bayésiennes (Caraux & Lechevallier 1996)

Pour les coraux, qui nous intéressent plus particulièrement, on obtient approximativement entre **60%** et **80%** de résultats d'identification corrects. D'après l'interprétation des experts, ces résultats ne sont pas surprenants. Ils sont assez proches des pourcentages calculés lors d'une identification manuelle, c'est-à-dire en parcourant l'arbre et en renseignant les caractères un à un. Ces résultats ne sont pas mauvais. Dans le domaine des coraux, un pourcentage de réussite proche de 80% est déjà un « bon résultat ».

4.9.4 Expérience 2 : en utilisant les structures

L'expérience 2 montre que les résultats du tableau précédent (4.1) peuvent être améliorés, en utilisant la structuration du domaine (le modèle) et les hiérarchies de valeurs (attribut taxonomique). Les bases de l'UCI étant non structurées, nous ne les avons pas inclus dans le tableau comparatif.

Dans cette expérience, les résultats obtenus sont exprimés par un intervalle. La borne inférieure de l'intervalle correspond au pourcentage de bon classement le plus pessimiste, la borne supérieure, au plus optimiste.

Classement pessimiste et optimiste

Dans le cas des 4 bases présentées, la classe a déterminer est exprimée par l'attribut de classe *taxon* qui est de type hiérarchique. Ainsi, le classement pessimiste est obtenu en comptabilisant tout mauvais classement sans considérer les proximités des classes.

Par exemple, étant donnée une description pré-classifiée ω associée à la classe *Pocillopora Damicornis bulbosa*, un écomorphe de l'espèce *Pocillopora Damicornis*. L'identification automatique de ω, sans considérer sa classe d'appartenance, conduit à associer cette description à l'espèce *Pocillopora Damicornis*. Le résultat est donc dans l'absolu « presque » correcte, mais une erreur entière (1) est comptabilisée.

Le pourcentage optimiste tient compte de la distance entre la classe trouvée et la classe réelle. Cette distance est une *distance d'arbre* ($0 \leq d \leq 1$, avec $d \in \mathbb{R}$), calculée par la mesure de dissimilarité introduite au chapitre suivant.

Bases structurées	Cas	Hiérarchiques	Composants	**Résultats**
Pocilloporidae (coraux)	113	12	48	[63% 79%]
Siderastreidae (coraux)	60	5	27	[80% 84%]
Fungiidae (coraux)	63	5	31	[83% 92%]
Hyalonema (éponges)	60	7	36	[59% 75%]

TAB. 4.2 – *Résultat d'identification monothétique par validation croisée, en utilisant la structure du domaine et les hiérarchies de valeurs.*

Interprétation

Cette expérience illustre l'augmentation de la qualité des résultats d'identification obtenus, grâce à l'utilisation de la structure du domaine et des descripteurs hiérarchiques. Même dans le

cas pessismiste, l'augmentation est de 2% ou 3% en moyenne par rapport à la même expérience effectuée sans utilisation des structures. Notons que dans ce dernier cas, les résultats sonr également pessimistes, étant donné que la structure de l'attribut de classe n'est pas prise en compte. On ne peut donc calculer de distances entre la classe trouvée et la classe définie.

Dans le cas optimiste, le pourcentage dépasse les **75%** dans tous les cas et est proche de **80%** en moyenne. Ce qui est un bon résultat, étant donné la complexité et la fréquence des valeurs non-renseignées dans les données sur les coraux et les éponges marines.

4.9.5 Conclusion sur les expérimentations

Etant donné une base de connaissance BC, le même type d'expérimentation peut être conduit très simplement avec $\mathcal{IKBS}$. Les résultats obtenus donnent une bonne idée de la qualité de la base, l'exhaustivité des classes représentées, etc. Une analyse approfondie peut amener à remarquer que les erreurs de classement concernent d'avantage une ou plusieurs espèces de la base parmis les autres. L'expert peut se fonder sur ces résultats pour décider d'ajouter de nouvelles descriptions, manquants dand l'échantillonage initial.

4.10 Conclusion

Nous avons exposé une méthode d'identification d'objets biologiques complexes représentés dans le formalisme CoDesc, permettant de construire des arbres d'identification dynamiques, en mettant en oeuvre la méthode *expérimentale*. Les relations de dépendance entre descripteurs sont prises en compte, ce que Conruyt (1994) considère comme une condition fondamentale de la « robustesse » dans les systèmes d'aide à la détermination et à la classification des objets du vivants. La méthode donne la possibilité d'effectuer l'identification d'un spécimen, « du niveau le plus général au plus spécifique », ce qui permet d'appréhender de manière logique la « taxonomie des Espèces ». De plus, la notion de poids associé au descripteur permet d'introduire des connaissances d'ordre stratégique pour la construction des clés et la possibilité pour l'expert de construire de manière semi-automatique une clé d'identification qui reflète de la meilleure façon sa vision du domaine. Enfin, cette méthode d'identification est accessible par le Web via un simple navigateur, pour les experts du domaine, les amateurs ou les techniciens de l'environnement. Les expériences exposées illustrent l'intérêt de la structuration du domaine pour augmenter la qualité des résultats d'identification trouvés.

Cette méthode d'identification a été adoptée par les experts qui ont développés la « base de connaissances sur les coraux des Mascareignes », comme en témoigne G. Faure dans l'extrait suivant :

> La finalité première de la « Base de Connaissances Coraux des Mascareignes » est de pouvoir identifier les différentes espèces de Coraux présentes dans l'Archipel (186 espèces). Dans ce dessein, la Base propose un « Arbre d'identification » dont le cheminement s'effectue au moyen d'une suite de questions-réponses au niveau générique puis spécifique, et dont le choix est favorisé par une ou plusieurs illustrations. Par ailleurs, un glossaire est toujours disponible ainsi que des commentaires, permettant

aux non-initiés de se familiariser avec la terminologie des caractéristiques écologiques et morphologiques des espèces. Chaque réponse peut :

- soit apporter la solution à l'identification spécifique de l'échantillon,
- soit générer une autre question plus précise ... et ainsi de suite jusqu'à l'obtention de l'identification de l'échantillon.

Par ailleurs, si l'utilisateur au cours de son cheminement éprouve des difficultés à choisir une réponse parmi celles proposées pour chacune des questions, il lui est toujours possible de ne pas répondre en renseignant la valeur *inconnu*. Ce faisant il génère alors une autre question.

De plus, en cas de doute concernant une réponse antérieurement apportée, il peut revenir en arrière dans le cheminement déjà effectué. Le nombre de questions-réponses utilisées varie en fonction de la fiabilité de l'arbre d'identification, de l'identité de l'espèce ... Et de la qualité d'observation de l'utilisateur.

La méthode offre par ailleurs la possibilité de coefficienter les caractéristiques (objets et attributs) au cours de l'édification du modèle, permettant ainsi au Systématicien de hiérarchiser et de mettre en exergue les caractères les plus significatifs et/ou déterminants. Cette hiérarchisation se traduit par une réduction du nombre de questions-réponses au niveau l'arbre d'identification, et donc d'accéder le plus rapidement possible à l'identification de l'espèce considérée.

Il est en outre possible de faire apparaître pour chaque constituant du modèle ainsi que de la base de cas et de l'arbre decision, des commentaires et des illustrations qui constituent une orientation et une aide à l'identification. Le recours à la classification engendrée par l'outil apporte à l'expert, soit une confirmation, soit une interrogation sur l'existence même d'espèces affines. Il permet également de remettre éventuellement en cause la validité des identifications lorsque par exemple plusieurs cas (spécimens) rapportés à une seule et même espèce se repartissent dans plusieurs classes.

Ainsi, l'outil $\mathcal{IKBS}$ adapté aux coraux apporte une fiabilité dans l'identification des espèces, que n'autorisent pas les moyens classiques utilisés jusqu'à présent : *identification par analogies* [97] photographiques, descriptions narratives, clefs de détermination ... Si cependant, compte tenu de la variabilité intraspécifique voire intra-coloniale qui caractérise les Coraux, cette fiabilité ne peut être totale, l'outil $\mathcal{IKBS}$ permet :

- d'alerter le Systématicien sur la validité de ses propres identifications,
- de pouvoir modifier à tout moment le modèle et l'arbre d'identification,
- de remettre en cause ou de confirmer la validité d'espèces voisines.

97. Nous traiterons de cet aspect au chapitre suivant.

Chapitre 5

Comparaison et classification d'objets complexes

Sommaire

5.1 Introduction

Nous souhaitons pouvoir **comparer** des descriptions d'objets complexes représentées dans le formalisme CoDesc, afin de pouvoir les regrouper selon leurs ressemblances ou bien déterminer l'ensemble des descriptions qui sont les plus proches d'une description particulière. Nous rappelons que ces descriptions sont instances de modèles descriptifs (les concepts du domaine) qui représentent des taxons de haut niveau (famille ou genre). Ceux-ci imposent une structure modulaire aux descriptions, les éléments de leur extension. Les instances sont décrites par des types complexes de valeurs (ensemble, conjonction, disjonction, intervalle, inconnu, etc.), les éléments du co-domaine des attributs définis au sein des modèles descriptifs (cf. chap. 2).

La possibilité de pouvoir comparer deux à deux les objets permet en particulier de mettre en place une stratégie d'identification dite *polythétique*, car elle prend en compte de manière globale l'intégralité des caractéristique des individus décrits[98]. Cette méthode présente de meilleure performance pour l'identification, comme nous l'illustrerons dans la suite de cet exposé.

Pour comparer deux objets, il est nécessaire de disposer d'une mesure métrique qui caractérise précisément les ressemblances (similarité) ou bien les dissemblances (dissimilarité) des deux descriptions de ces objets.

Définition 5.1 *dissimilarité, distance et normalisation*

Soit un ensemble de descriptions Ω*, une mesure de dissimilarité est une fonction*

$d : \Omega \times \Omega \rightarrow \mathbb{R}^+$

respectant les propriétés suivantes :

$d(x,y) = 0 \Leftrightarrow x = y$ *(minimalité)*

$d(x,y) = d(y,x)$ *(symétrie)*

Une distance est une mesure de dissimilarité respectant en plus l'inégalité triangulaire :

$d(x,y) + d(y,z) \geq d(x,z)$

Elle est dite normalisée si son co-domaine est l'intervalle [0 1].

La mesure que nous proposons est une *dissimilarité normalisée.*

5.1.1 Contexte général

En Analyse de Données

Les mesures de *similarité* ou de *proximité* jouent un rôle fondamental dans les méthodes de classification, de reconnaissance des formes, de raisonnement par cas, et plus généralement, en Analyse des Données. Les distances sur des objets caractérisés par des variables numériques sont bien connues et ont été largement étudiées. Cependant, dans les applications réelles, il est courant de mesurer sur un objet des variables de type quantitatif, qualitatif (*données hétérogènes*) dont les valeurs peuvent être manquantes, inconnues ou multiples (intervalles numériques ou ensembles finis de valeurs). Pour de telles données, les distances usuelles ne peuvent pas

98. L'approche exposée au chapitre précédent (chap. 5) est dite *monothétique*, car elle ne considère qu'un seul caractère à la fois, successivement.

s'appliquer. L'approche habituelle en Analyse de Données, pour contourner la difficulté liée à la présence de valeurs de type ensemble est de coder de manière *ad hoc* les valeurs pour faire entrer ces valeurs complexes dans un cadre numérique, ce qui conduit à une perte relativement importante d'information.

Concernant les dissimilarités sur des données comportant des valeurs de type intervalle, différentes propositions ont été faites, notamment par Ichino (1986), Ichino (1988) et Gowda & Diday (1991). Ces derniers proposent une distance qui fait intervenir la « position », « l'étendue » et le « contenu » des descriptions des objets. Les mesures de dissimilarité proposées par ces auteurs sont efficaces dans la plupart des cas, mais ne distinguent pas certaines situations pourtant très différentes (Polaillon 1998). De plus, elles traitent toutes les valeurs descriptives des variables quantitatives comme des intervalles, et ne prennent donc pas en compte le caractère discret des ensembles finis de valeurs.

En Analyse de Données Symboliques

Plus récemment, en Analyse de Données Symboliques, une famille de mesures de dissimilarités sur les Objets Symboliques (cf. 1.9.5) a été proposé par Carvalho (1998). Ces mesures se fondent sur deux opérations dyadiques sur les objets symboliques, l'union et l'intersection, ainsi que sur une fonction monoadique appelée *potentiel descriptif*. La dissimilarité de deux objets symboliques est exprimée en fonction du « volume » de leur union et intersection (Ichino 1994).

Dans les modèles à objets

Parallèlement, dans les représentations dites « à objets », différentes mesures sont proposées (Bisson 1995, Valtchev & Euzenat 1997, Valtchev 1999) visant à extraire des ressemblances entre instances ou entre classes, en tenant compte de toute la complexité relative à la structure de ces entités. Différents problèmes sont abordés comme la prise en compte des *collections d'objets*, des relations et de problèmes liés à la *circularité entre objets*, la structure sous-jacente étant un graphe.

5.1.2 Dans notre contexte

Les descriptions que nous souhaitons comparer sont caractérisées par des valeurs complexes (ensembles, intervalles, valeurs hiérarchiques, etc.) et sont munis d'une structure arborescente induite par la donnée d'un modèle descriptif, dont l'extension les recouvre tous. En effet, nous considérons que la donnée d'un seul modèle descriptif (concept) est suffisante pour caractériser un ensemble de descriptions (noté X par la suite), comme nous l'avons illustré au § 3.4 et au § 4.5.5 pour traiter le problème de l'identification d'objets situés dans une hiérarchie $\mathcal{H}$ de concepts.

Dans ce contexte, les descriptions n'entretiennent pas de relations entre elles, elles sont indépendantes les unes des autres, mais sont conformes à la structure du modèle descriptif auquel elles appartiennent (modulaire du fait des objets *absent possible*). Le problème de la mise au point d'une mesure numérique permettant de caractériser les différences entre les descriptions se pose donc d'une manière quelque peu différente que dans le cas des modèles à objets (notamment pour les relations) et rejoint davantage l'approche développée pour les objets symboliques.

Les problèmes qui se posent particulièrement dans notre contexte sont :

- Comment normaliser la mesure, de manière à ce que chaque élément de description joue un rôle identique ?
- Comment prendre en compte la « longueur » des valeurs de type ensemble comportant à la fois des intervalles et des valeurs discrètes, comme par exemple l'ensemble {1, [5 7], 12} ?
- Comment interpréter les relations d'ordre entre valeurs, celles-ci étant des éléments du treillis de valeur (co-domaine) associé au domaine de chaque attribut ?
- De quelle manière tenir compte de la structure des descriptions dans la mesure ?

Plan de l'exposé

Nous proposons dans un premier temps, une mesure de *dissimilarité locale* permettant de comparer de manière très fine des objets caractérisés par des variables de différents types : numérique, symbolique, et hiérarchisé. Les valeurs des variables peuvent être des ensembles finis de valeurs discrètes et/ou d'intervalles. Après avoir introduit la mesure locale, nous la comparons sur des exemples aux mesures proposées par Hausdorff, Gowda et Diday (Gowda & Diday 1991) et de Ichino (Ichino 1986), puis nous l'utilisons dans une méthode de classification ascendante s'appuyant sur un opérateur d'union cartésienne.

Dans un second temps, nous développons une mesure de *dissimilarité globale* permettant de tenir compte de la structure de chaque observation. Nous proposons ensuite une mesure de *dissimilarité générale* permettant de combiner la mesure locale et la mesure globale pour comparer des objets complexes en tenant compte de toute l'information contenue dans les descriptions.

Nous présentons ensuite plusieurs méthodes de classification et d'identification fondées sur la mesure proposée. Toutes ces méthodes sont implantées dans l'application $\mathcal{IKBS}$.

Enfin, nous illustrons l'intérêt de cette mesure pour la mise en oeuvre d'une méthode d'identification *polythétique* à l'aide de deux expérimentations.

5.2 Mesure de dissimilarité locale

Dans Diatta et al. (1999), nous avons proposé la mesure de dissimilarité locale, nous en présentons ici les principales caractéristiques.

Normalisation

La définition d'une similarité ou de dissimilarité sur des variables hétérogènes (de différents types) pose le problème de la *normalisation*.

Comment procéder pour que chaque variable joue un rôle identique dans la mesure de la distance entre deux objets ?

La difficulté est d'autant plus grande que nous nous autorisons la présence simultanée de variables monovaluées et multivaluées quel que soit le type. Pour cela, nous avons considéré une valeur v d'une variable $A \in \mathcal{A}$, $\mathcal{A}$ étant l'ensemble des variables définies au sein d'un modèle descriptif considéré, comme une partie de son domaine (cf.§ 2.4). Nous définissons une distance

normalisée unique sur chaque variable à partir d'un indice *taille*, noté $s(v)$ adapté à chaque type de variable. Cet indice, ainsi que d'autres sont introduits ci-dessous.

5.2.1 Les indices de base

Dans la suite v (au lieu de v^A pour simplifier la notation) désignera une valeur d'une variable A.

Bornes et extrémités

Supposons que A soit une variable quantitative. Si v est un intervalle alors v_{lb} et v_{ub} représentent les bornes repsectivement inférieure et supérieure de v et $\mathbf{b}(v) = \{v_{lb}, v_{ub}\}$ l'ensemble des points extrémités de l'intervalle v. Si v est de type ensemble, alors $\mathbf{b}(v) = \cup_{e \in v} \mathbf{b}(e)$, c'est-à-dire l'ensemble des extrémités contenues dans v.

Enveloppe convexe

L'enveloppe convexe de v est tout simplement l'intervalle $\mathbf{co}(v) = [v_{lb}\ v_{ub}]$.

Mesure de Lebesgue

Soit $\mathbf{L}$ la *mesure de Lebesgue* ou en abrégé *L-mesure* sur $\mathbf{dom}(A)$. Il s'agit d'une fonction réelle positive définie par :

$$\mathbf{L}(v) = \begin{cases} v_{ub} - v_{lb} & \text{si } v \text{ est précis ou un intervalle;} \\ \sum_{e \in v} \mathbf{L}(e) & \text{si } v \text{ est de type ensemble.} \end{cases}$$

5.2.2 « Longueur » et « taille » d'une valeur

L'approche habituelle pour prendre en compte la longueur d'une valeur v consiste à considérer soit le cardinal de v si c'est un ensemble de valeurs discrètes, soit l'enveloppe convexe de v, si v comporte des intervalles (Gowda & Diday 1991, Ichino 1986). Le problème dans ce cas est le suivant :

Deux valeurs sont considérées équivalentes du point de vue de la mesure, à partir du moment où leur enveloppe convexe est identique. Par exemple, $v_1 = \{1, [4\ 5], 6\}$ est considérée équivalente à $v_2 = \{1, 6\}$ ou encore à $v_3 = \{[1\ 6]\}$. Les dissimilarités exprimées sur les différents couples sont alors indifféremment nulles, v_1, v_2 et v_3 exprimant pourtant, dans la réalité, des situations bien différentes. En effet, v_3 exprime une *variation* numérique de 1 à 6, alors que v_2 exprime une *imprécision* entre 1 et 6.

Nous nous plaçons ici dans le cadre général des variables numériques. Le problème est similaire pour les attributs nominaux, ordinaux ou taxonomiques, en considérant la distinction entre les valeurs disjonctives (ou ensemble) qui traduisent une imprécision et les valeurs conjonctives qui traduisent une variation.

Etant donné X, un ensemble d'individus pour lesquels ont ne considère pas de structures (aucune relation entre descripteurs n'est considérée). X peut être représenté par un tableau de données symboliques. On notera x ou y les éléments de X, pour les différencier des éléments $\omega \in \Omega$, les descriptions complexes munies de structures.

Pour calculer la valeur de v, nous effectuons la distinction entre d'une part les valeurs de type ensemble (imprécision) et d'autre part les valeurs de type intervalle (variation).

Par exemple : soit $A \in \mathcal{A}$, de type numérique. $\{1, 5.8, 37\}$ est une valeur de type ensemble. Toute valeur numérique ($\in \mathbb{R}$) simple v est également considérée comme le singleton $\{v\}$. De même, pour les variables symboliques, toute valeur possible est considérée comme un ensemble (éventuellement l'ensemble vide pour la valeur *non renseignée*).

Soit $n_A = \max\{|v_x^A| : x \in X,\ v_x^A \text{ est de type ensemble}\}$, le *plus grand cardinal* des valeurs de A de type ensemble.

Soit $\lambda_A = \min\{\mathbf{L}(v_x^A) : x \in X, v_x^A \text{ est de type intervalle}\}$, *la plus petite L-mesure* des valeurs de A de type intervalle. Le paramètre λ_A est égal à 1 si A ne comporte pas de valeur de type intervalle.

On définit la *longueur* d'une valeur v par :

$$\mathbf{l}(v) = \begin{cases} 2\,\mathbf{L}(v) & \text{si } v \text{ est de type intervalle;} \\ \mathbf{L}(v) + \frac{\lambda_A\,|v|\,\mathbf{L}(\mathbf{co}(v))}{2\,n_A\,\mathbf{L}(\mathbf{dom}(A))} & \text{si } v \text{ est de type ensemble.} \end{cases}$$

Ces indices ont été définis pour mesurer le « contenu » des diverses valeurs. La mesure de Lebesgue aurait été un bon candidat si nous ne traitions que des valeurs de type intervalle. Mais cette mesure n'est pas adaptée pour les ensembles discrets sur lesquels elle est indifféremment nulle. Pour une valeur qui n'est pas un intervalle, on considère la L-mesure $\mathbf{L}(co(v)))$ de son enveloppe convexe qui est normalisé par $\mathbf{L}(dom(A))$.

Malgré cette normalisation, il est souhaitable qu'une valeur de type ensemble ait une plus grande étendue qu'un intervalle, si une valeur intervalle est présente dans les données. D'où la normalisation par par la L-mesure de la plus petite valeur λ_A pour tenir compte de la présence éventuelle d'une valeur intervalle.

Cependant, concernant l'étendue, on doit pouvoir être capable de distinguer entre les valeurs $\{b, [c\ d], e\}$, $\{b, c, e\}$ et $\{[b\ c], d, e\}$. C'est pourquoi, pour une valeur de type ensemble, on ajoute la L-mesure de ses elements, et la longueur d'un intervalle est égale à deux fois sa L-mesure.

Taille d'une valeur

Nous définissons la *taille* d'une valeur v par :

$$\mathbf{s}(v) = \begin{cases} \mathbf{l}(v) & \text{si } A \text{ est de type quantitative} \\ |v| & \text{si } A \text{ est qualitative (nominale, ordinale ou hiérarchique)} \end{cases}$$

5.2.3 Définition de la mesure de dissimilarité locale

Nous définissons la dissimilarité entre deux descriptions x et y de X, comme la somme pondérée des dissimilarités locales sur chacun des attributs, par :

$$D(x,y) = \left(\sum_{i=1}^{i=p} \beta_i \, d^r(v_x^{A_i}, v_y^{A_i}) \right)^{\frac{1}{r}} \tag{5.1}$$

où $r \geq 1$ et où $\beta_i \geq 0$ est le poids de la variable A_i, $i = 1, \ldots, p$.

Dans la pratique, la valeur du paramètre r sera souvent 1, 2 ou ∞.

Dissimilarité sur une variable

Pour une variable A, la dissimilarité $d(v_x^A, v_y^A)$ est définie comme la combinaison convexe de deux composants d_P et d_C relatifs à la *position* relative des valeurs et de leur *contenu* par :

$$d(v_x^A, v_y^A) = \eta \, d_P(v_x^A, v_y^A) + \zeta \, d_C(v_x^A, v_y^A) \tag{5.2}$$

où $\eta, \zeta \geq 0$, $\eta + \zeta = 1$. η et ζ permettent de moduler l'importance relative de d_P et d_C. Par exemple, dans le cas où $\eta = 0$, la mesure globale ne considère que la donnée du contenu.

Dissimilarité fondée sur la position

La position relative de deux valeurs est mesurée à l'aide de la dissimilarité d_P calculée sur les valeurs réelles moyennes par :

$$d_P(v_x^A, v_y^A) = \frac{1}{\mathbf{s}(\mathbf{dom}(A))} \left| \frac{1}{|\mathbf{b}(v_x^A)|} \sum_{\alpha \in \mathbf{b}(v_x^A)} \alpha \; - \; \frac{1}{|\mathbf{b}(v_y^A)|} \sum_{\alpha \in \mathbf{b}(v_y^A)} \alpha \right|. \tag{5.3}$$

Dissimilarité fondée sur le contenu

La composante d_C de la dissimilarité basée sur le contenu est définie comme la moyenne géométrique de deux termes d_{SP} et d_{CP}, relatifs aux parties *spécifiques* et à la partie *commune* des valeurs, respectivement :

$$d_C(v_x^A, v_y^A) = \sqrt{d_{SP}(v_x^A, v_y^A) * d_{CP}(v_x^A, v_y^A)} \tag{5.4}$$

La composante relative aux parties spécifiques est :

$$d_{SP}(v_x^A, v_y^A) = \frac{1}{\mathbf{s}(\mathbf{dom}(A))} \, \mathbf{s}(v_x^A \triangle v_y^A). \tag{5.5}$$

où $v_x^A \triangle v_y^A$ est la différence symétrique entre les valeurs v_x^A et v_y^A.

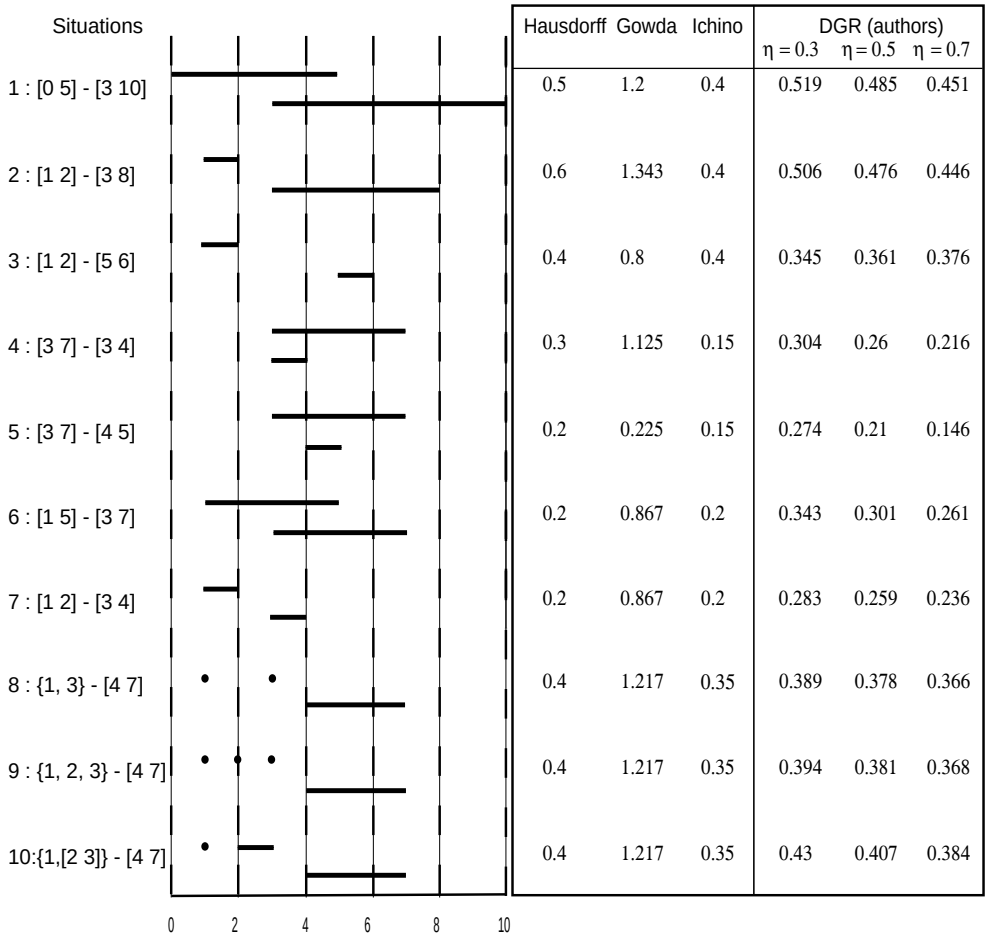
La composante relative à la partie commune est :

$$d_{CP}(v_x^A,v_y^A) = 1 - \frac{1}{\mathbf{s}(\mathbf{dom}(A))}\,\mathbf{s}(v_x^A \cap v_y^A) \tag{5.6}$$

Remarquons que d_{CP} est en fait une *pseudo-dissimilarité* car $d_{CP}(v,v)$ peut ne pas être égale à 0 pour certaines valeurs de v. Pour une valeur donnée de $d_{SP}(v_1,v_2)$, plus la partie commune est importante entre v_1 et v_2, plus la valeur de d_C est petite. De même, pour une valeur donnée de $d_{CP}(v_1,v_2)$, plus la partie spécifique est grande entre v_1 et v_2 plus la valeur de d_C est importante.

5.2.4 Comparaison avec d'autres travaux

Nous présentons maintenant une comparaison de la dissimilarité locale proposée avec celles de Gowda et Diday (Gowda & Diday 1991), de Hausdorff et de Ichino (Ichino 1986). Le tableau 5.1 est inspiré de (Polaillon 1998) où peut être trouvée une étude comparative des trois dernières mesures de dissimilarité.



	Hausdorff	Gowda	Ichino	DGR (authors) η = 0.3	η = 0.5	η = 0.7
1 : [0 5] - [3 10]	0.5	1.2	0.4	0.519	0.485	0.451
2 : [1 2] - [3 8]	0.6	1.343	0.4	0.506	0.476	0.446
3 : [1 2] - [5 6]	0.4	0.8	0.4	0.345	0.361	0.376
4 : [3 7] - [3 4]	0.3	1.125	0.15	0.304	0.26	0.216
5 : [3 7] - [4 5]	0.2	0.225	0.15	0.274	0.21	0.146
6 : [1 5] - [3 7]	0.2	0.867	0.2	0.343	0.301	0.261
7 : [1 2] - [3 4]	0.2	0.867	0.2	0.283	0.259	0.236
8 : {1, 3} - [4 7]	0.4	1.217	0.35	0.389	0.378	0.366
9 : {1, 2, 3} - [4 7]	0.4	1.217	0.35	0.394	0.381	0.368
10:{1,[2 3]} - [4 7]	0.4	1.217	0.35	0.43	0.407	0.384

FIG. 5.1 – *Comparaison de mesures de dissimilarités locales.*

Notons qu'à travers l'étude de Polaillon (1998), la mesure de Ichino (Ichino 1986) s'est avérée la plus satisfaisante des trois mesures : Hausdorff, Gowda et Ichino.

L'expérience porte sur une base de 10 objets décrits par une seule variable quantitative dont les valeurs sont de type intervalle ou ensemble. Il s'agit de voir comment les différentes mesures se comportent en fonction des positions relatives et des étendues respectives des valeurs.

Chaque situation représente les valeurs prises par la variable de description sur deux objets. Les deux valeurs sont séparées par un tiret. En ce qui concerne la mesure que nous proposons, les dissimilarités sont calculées, pour différentes valeurs du paramètre η (trois dernières colonnes), en attribuant un poids unitaire à la variable descriptive.

Dans les situations 4 et 5, les parties commune et spécifiques sont identiques mais les positions relatives des valeurs sont manifestement différentes. Ces deux situations montrent l'insensibilité de la mesure de Ichino par rapport aux positions relatives. Les mesure de Hausdorff, Ichino et Gowda et Diday ne distinguent pas les situations 6 et 7. Ces deux situations sont clairement différentes, étant donné que les deux valeurs de la situation 6 ont une partie commune non vide contrairement à celles de la situation 7.

Dans les situations 8, 9 et 10, le même objet (de description [4 7]) est respectivement comparé avec trois objets de descriptions différentes. Les mesures de Hausdorff, Gowda et Ichino ne distinguent pas ces trois situations du fait qu'elles traitent les valeurs des variables quantitatives comme leurs enveloppes convexes. Notons que la mesure que nous proposons est sensible à la faible différence entre les situations 8 et 9.

Ces trois dernières situations illustrent bien le fait que la mesure que nous proposons tient compte aussi bien de la cardinalité des valeurs de type ensemble que de l'étendue des éléments de ces valeurs.

5.3 Applications

5.3.1 Classification Ascendante Hiérarchique (CAH)

(Cette méthode a été réalisée par D. Grosser et J. Diatta)

A partir d'un ensemble X de n descriptions, l'algorithme de CAH construit une hiérarchie de classes, par regroupement deux à deux des descriptions les plus proches, selon le critère donné par la mesure de dissimilarité présentée. Chaque nouvelle classe est obtenue à l'aide d'un *opérateur d'union cartésienne*, « $\uplus$ », défini, pour tout x,$y \in X$, par $x \uplus y = \left(V_x^{a_1} \cup V_y^{a_1}\right) \times \cdots \times \left(V_x^{a_p} \cup V_y^{a_p}\right)$. La méthode est *ascendante* du type de l'*algorithme de Ward*, où chaque classe devient une nouvelle description, *objet composite*, obtenu par union cartésienne des descriptions qui la composent. Par analogie avec l'algorithme de Ward, cet objet composite joue le rôle du centre de gravité de la classe, le poids de cette dernière étant le nombre de descriptions qui la composent. Les classes sont ainsi exprimées dans le même formalisme que les individus, et correspondent ainsi à une *description synthétique*. De ce fait, la classe obtenue peut être comparée aux autres descriptions.

Le processus est réitéré jusqu'à l'obtention d'une classe unique ou bien lorsque le nombre minimal de classes (>1) est atteint (celui-ci est paramétré avant l'exécution de la méthode). Des relations sont établies entre chaques classes et les éléments qui la composent, de manière à former un arbre appelé *Dendrogramme* dont les feuilles sont les éléments initiaux de X.

5.3.2 CAH sur des données simples

Pour illustrer notre approche, nous l'appliquons sur un petit jeu de données facile à interpréter, des données relatives à des graisses et des huiles (Gowda & Diday 1991, Ichino 1986, Polaillon 1998).

Sur celles-ci sont mesurées quatre variables quantitatives : *Specific gravity, freezing point, io. value (indice d'iode) et sa. value (indice de saponification)* dont les valeurs sont des intervalles, et une variable qualitative : *m.f. acids (acides aminés).*

label	sample name	sp. gravity (g/cm^3)	fr.pt (^oC)	io. value	sa. value	m.f. acids
0	Linseed oil	0.930–0.935	-27 to -8	170–204	118–196	L,Ln,O,P,M
1	Perilla oil	0.930–0.937	-5 to -4	192–208	188–197	L,Ln,O,P,S
2	Cotton-seed	0.916–0.918	-6 to -1	99–113	189–198	L,O,P,M,S
3	Sesame oil	0.920–0.926	-6 to -4	104–116	187–193	L,O,P,S,A
4	Camellia	0.916–0.917	-21 to -15	80–82	189–193	L,O
5	Olive oil	0.914–0.919	0 to 6	79–90	187–196	L,O,P,S
6	Beef-tallow	0.860–0.870	30 to 38	40–48	190–199	O,P,M,S,C
7	Lard	0.858–0.864	22 to 32	53–77	190–202	L,O,P,M,S,Lu

TAB. 5.1 – *Données sur des graisses et huiles*

Le nombre de classes de la partition est déterminé par l'étude de l'histogramme des valeurs de l'indice d'agrégation de la méthode hiérarchique. Un saut significatif est constaté pour un nombre de classes égal à 3.

Remarquons que les objets composites formés à chaque étape de l'algorithme sont bien l'union cartésienne des objets qui les composent, même si les classes (objets composites finaux) sont affichées avec les valeurs de variables quantitatives remplacées par leurs enveloppes convexes. Par exemple, la classe 2 est composée de deux graisses : *beef-tallow* et *Lard*, la valeur d'indice d'iode est comprise entre 40 et 77 (le calcul de la dissimilarité est faite sur la paire d'intervalles {[40 48],[53 77]}). Pour ces observations, chaque acide de { O,P,M,S,C,L,Lu } est présent dans au moins une de ces graisses .

Cluster	samples	sp. gr.	fr.pt (^oC)	io. value	sa. value	m.f. acids
1	0,1	0.930 – 0.937	-27 to -4	170 – 208	118–197	L,Ln,O,P,M,S
2	6,7	0.858 – 0.870	22 to 38	40 – 77	190–202	L,O,P,M,S,C,Lu
3	2,3,4,5	0.914 – 0.926	-21 to 6	79 – 116	187–198	L,O,P,M,S,A

TAB. 5.2 – *Description des classes de la partition*

5.4 Avantages de la mesure locale

Nous avons proposé une mesure de dissimilarité applicable sur des données mixtes, quantitatives (valeurs discrètes ou de type intervalle) et qualitative (valeurs symbolique) permettant

de comparer de manière très fine des descriptions caractérisées par de telles valeurs. En résumé, cette mesure présente les avantages suivants :

- Traitement dans un cadre unique de variables de différents types ;
- Possibilité de moduler l'importance de la position et du contenu des valeurs descriptives ;
- Différenciation de situations que les autres approches ne distinguent pas ;

De plus, les valeurs inconnues et manquantes (ou inapplicables) sont traitées naturellement du fait que dans notre approche, un attribut est considéré comme étant une application à valeurs dans l'ensemble des parties de son domaine.

5.5 Mesure de dissimilarité pour des données structurées

Dans la plupart des méthodes traditionnelles, les variables sont considérées indépendamment les unes des autres. Nous proposons une méthode mathématique permettant de prendre en compte les « connaissances de fond » et les relations entre composants et entre variables utilisés pour la description des objets complexes (les descriptions).

Nous considérons pour cela un modèle mathématique général, pour lequel l'aspect structurel est mis en avant, indépendamment d'autres considérations liées à la représentation des connaissances, telles que nous les avons détaillées au chapitre 2.

5.5.1 Modèle et description structurée

Nous considérons dans ce contexte la notion de *modèle structuré*.

Définition 5.2 *Un modèle structuré est un ensemble partiellement ordonné de n éléments, notés $\mathcal{C}$, où chaque élément $E \in \mathcal{C}$ (concept) est un composant caractérisé par un ensemble fini d'attributs. L'ensemble des attributs associés à E est noté A_E (cf. 2.4).*

Et celle de description structurée :

Définition 5.3 *Une description structurée est une instance $\omega \in \Omega$ d'un sous-ensemble P d'éléments de $\mathcal{C}$ (certains composants E peuvent être absents), où pour chaque attribut A_E, $E \in P$, est assignée une valeur, notée $v_\omega^{A_E}$. Les composants de P sont dits présents sur ω. Ceux de l'ensemble $\mathcal{C} \setminus P$ sont dits absents de ω.*

5.5.2 Description globale

La description structurée globale d'un individu est ainsi basée sur :

- la structure ordonnée de $\mathcal{C}$,
- la présence / absence de certains composants.

Afin de prendre en compte la structure d'ordre (induite par la relation de composition) de $\mathcal{C}$, nous considérons la fonction suivante, appelée *index de filiation* associée à $\mathcal{C}$:

Définition 5.4 *Index de filiation*

$\phi : \mathcal{C} \rightarrow \mathbb{R}$, *définie par :*

$$\phi(E) = \begin{cases} 1 & \textit{si } E \textit{ est maximale} \\ \\ \underset{Y \in Pred(E)}{min} \frac{\phi(Y)}{|Suc(Y)|} & \textit{sinon} \end{cases}$$

La figure 5.2 illustre un exemple de modèle structuré sous forme de graphe, la liste des prédecesseurs et des successeurs associées, ainsi que l'index de filiation. Ce modèle est applicable dans le cas des descriptions structurées représentées par des arbres de description.

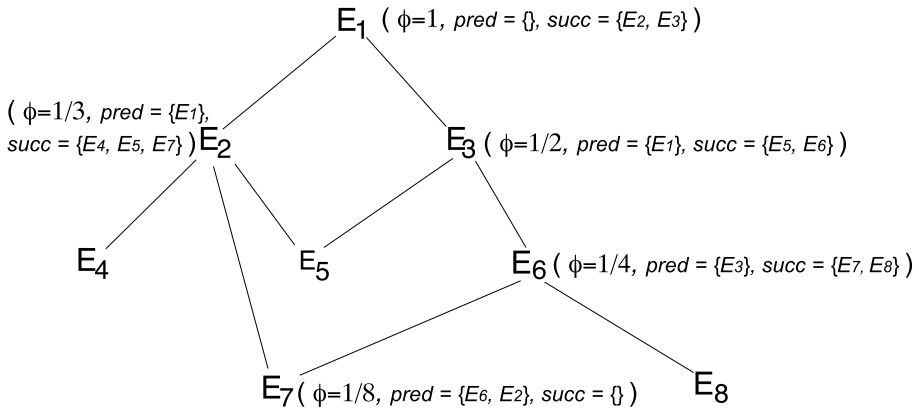


FIG. 5.2 – *Exemple de modèle structuré. Un index de filiation ϕ est associé à chaque objet $E \in \mathcal{P}$, ainsi que la liste des successeurs et prédecesseurs.*

Nous considérons également les situations où aucune information n'est disponible sur la présence / absence des objets, pour un individu particulier. Ces objets sont dits *inconnus* pour l'individu correspondant. Si un objet E est inconnu, pour $\omega \in \Omega$, $p_\omega(E)$ dénote la probabilité pour E d'être présent dans ω. En effet, même si la donnée est manquante, dans la réalité, le composant est nécessairement absent ou présent.

La description globale d'un individu ω peut-être identifiée, indépendamment des relations entre objets, par le n-vecteur $\big(\phi(E)\,\chi_\omega(E)\big)_{E \in \mathcal{C}}$, où χ_ω est défini sur E par :

$$\chi_\omega(E) = \begin{cases} 1 & \text{si } E \text{ est présent dans } \omega \\ 0 & \text{if } E \text{ est absent de } \omega \\ p_i(E) & \text{si } E \text{ est inconnu dans } \omega \end{cases}$$

5.5.3 Mesure de dissimilarité globale

A partir de ces définitions, la mesure de dissimilarité globale que nous proposons, notée D_G, consiste en la transformée de Minkowski sur le n-vecteur caractérisant uniquement la partie structurelle des descriptions :

$$D_G(\omega_1,\omega_2) = \left(\sum_{E \in \mathcal{C}} \phi(E)^r \left| \chi_{\omega_1}(E) - \chi_{\omega_2}(E) \right|^r \right)^{\frac{1}{r}}, r \geq 1. \quad (5.7)$$

Il est possible, sans modifier cette expression globale, d'élaborer d'autres indices qui caractérisent l'absence ou la présence des composants dans une description, ainsi que l'inconnu.

5.5.4 Combinaison de la mesure locale et globale

La mesure de dissimilarité que nous proposons est la transformée de Minkowski d'un 2-vecteur. Les composantes de ce vecteur sont la mesure de dissimilarité globale normalisée, D_G et la mesure de dissimilarité locale normalisée, D_L :

$$D(\omega_1,\omega_2) = \left(\left(\mu\, D_G(\omega_1,\omega_2)\right)^r + \left(\nu\, D_L(\omega_1,\omega_2)\right)^r \right)^{\frac{1}{r}}, r \geq 1 \quad (5.8)$$

μ and ν sont les coefficients de *normalisation*. Les applications exposées par la suite sont conduites avec $r = 1$. Lorsque la structure n'est pas considérée, sur des tableaux de données par exemple, le composant D_G est nul. Dans ce cas, l'expression est réduite à la partie locale ($\mu = 0$ et $\nu = 1$).

Notons que toute métrique peut être utilisée dans cette équation générale. La mesure de dissimilarité locale peut être par exemple la mesure Euclidienne, ou la mesure de dissimilarité locale exposée au § 5.2, ou encore une des mesures utilisées en apprentissage automatique, exposées dans Wilson & Martinez (1997) et applicables sur des données mixtes :

- Heterogeneous Value Difference Metric (HVDM),
- Discretized Value Difference Metric (DVDM),
- Windowed Value Difference Metric (WVDM).

Se référer à Wilson & Martinez (1997) pour l'exposé précis de ces différentes mesures, ainsi qu'une analyse détaillée de leur propriétés.

5.6 Méthodes de classification fondées sur la mesure globale

La mesure globale présentée ci-dessus est un outil de base pour différentes méthodes de classifications et d'identification implantées au sein de la plate-forme $\mathcal{IKBS}$:

1. Classification par méthodes des nuées dynamiques distribuées.
2. Classification conceptuelle.
3. Classification Ascendante Hiérarchique. La méthode a été exposée précédemment.
4. Identification *polythétique*.

L'exposé précis et détaillé de l'ensemble de ces méthodes sort du cadre de ce mémoire. Ces méthodes ont été adaptées à partir de méthodes classiques en Analyse de Données et en Apprentissage Automatique, pour prendre en compte la complexité des descriptions représentées dans le formalisme CoDesc.

Elles ont été réalisées pour la plupart par des étudiants de la maîtrise d'informatique, encadrés par H. Ralambondrainy (Pr.), J. Diatta (MCF) et D. Grosser (en thèse), de l'IREMIA de l'Université de la Réunion.

5.6.1 Nuées dynamiques distribuées à l'aide d'agents

Ce travail a été réalisé par (Vidot 1999), à partir de la méthode des « nuées dynamiques » (§ 1.9.3) de Diday et al. (1982) avec centres de gravités. L'originalité de cette méthode réside dans l'approche par agents proposée. Chaque donnée est considérée comme un agent [99]. Les principes de coopération et de négociation ont été abordés, mais n'ont pas été jugés pertinents pour le type de problème. Ce travail montre cependant de manière formelle que la distribution du calcul lors d'une classification ne modifie aucunement le résultat par rapport à l'approche séquentielle classique.

Les résultats n'ont pas été appliqués sur les bases coraux, car l'algorithme souffre encore de quelques lacunes. Il mériterait d'être révisé et amélioré. L'approche est cependant très intéressante.

5.6.2 Classification conceptuelle par hiérarchie faible conceptuelle

Ce travail a été réalisé par Soune-Seyne & Cadet (2001) et supervisé de manière approfondie par J. Diatta et D. Grosser.

Cette méthode de classification combine deux approches : l'approche conceptuelle et l'approche fondée sur les distances. La méthode produit une *hiérarchie faible* de concepts.

L'approche fondée sur les distances

L'avantage de cette approche réside essentiellement dans le fait que l'on dispose de résultats mathématiques qui établissent des bijections entre des systèmes de classification et des classes de distances. On passe alors d'un problème basé sur des ensembles de descriptions à un problème basé sur des distances entre individus. La méthode fournit une information sur la qualité des concepts obtenus à partir de l'extension, via un *indice d'isolation faible*.

L'inconvénient réside dans le fait qu'une partie de l'information est perdue sur la description des éléments constituant une classe, du fait que cette information a été transformée en une distance.

L'analyse conceptuelle formelle

L'analyse conceptuelle formelle construit un treillis, appelé treillis de concepts. L'avantage est que les concepts peuvent avoir plusieurs ascendants. Cependant, le nombre de concepts et de relations entre concepts peut être très élevé (croissance exponentielle), ce qui rend difficile la représentation et la manipulation.

99. Les agents sont les entités de base des Systèmes Multi-agents (Ferber 1989).

Hiérarchie faible conceptuelle

Le système classificatoire engendré par l'approche par hiérarchie faible conceptuelle est un bon compromis entre une hiérarchie et un treillis de concepts et présente un comportement intéressant du point de vue de la complexité (polynomial) et de la taille de la structure de graphe générée.

Nous conduisons actuellement un ensemble d'expérimentations à l'aide de cette méthode, sur les données de la « base de connaissances coraux ». L'approche nous semble très prometteuse.

5.6.3 Application de la Classification Ascendante Hiérarchique

Nous avons développé précédemment (§ 5.3.1), nous présentons maintenant une application de la CAH sur la famille des *Pocilloporidae*. La méthode est fondée sur l'utilisation la mesure de dissimilarité globale que nous avons proposée.

L'application suivante a été conduite à partir des descriptions de la famille *Pocilloporidae* de la BC coraux. Initialement, 114 descriptions de spécimens sont considérées. L'arbre taxonomique de cette famille (attribut de classe taxon) contient 19 noeuds : 4 genres, 10 espèces et 5 écomorphes.

Principe

Les 114 descriptions de spécimens sont synthétisés en 19 descriptions généralisantes, correspondant à chaque taxon. Celles-ci forment la partition initiale de la CAH. On applique ensuite la méthode précédente jusqu'à l'obtention d'une classe unique, synthèse des 19 classes initiales. L'indice de Ward permet de calculer la meilleur partition, en fonction de la variation de la dissimilarité des deux classes les plus proches. La meilleur partition calculée dans le cas de la famille *Pocilloporidae* est 10 classes.

La figure 5.3 illustre le Dendrogramme obtenu par CAH. Les classes initiales sont numérotées, ainsi que chaque étape de la classification. Nous n'avons pas indiqué dans cet exemple la valeur de la mesure pour chaque étape.

Interprétation

D'une façon générale, l'arbre de classification (ou dendrogramme) construit est intéressant de trois points de vue :

- La structure de l'arbre permet d'appréhender de manière globale les espèces affines, ainsi que les proximités et les distances entre les différentes classes construites. Dans l'exemple précédent, l'étape 7 associe les deux espèces *Pocillopora eydouxi* et *Pocillopora verrucosa* qui sont effectivement des espèces très proches (d'après G. Faure).
- A un niveau donné (nombre de classes fixées), l'arbre obtenu donne une vision particulière de la classification des différents taxons. Dans l'exemple précédent, la meilleure partition donne 10 classes. Une analyse approfondie devrait être conduite par les experts.
- La structure obtenue permet d'appréhender de manière globale les données présentes dans la BC, afin de mettre en évidence certaines incohérences ou au contraire mettre en avant

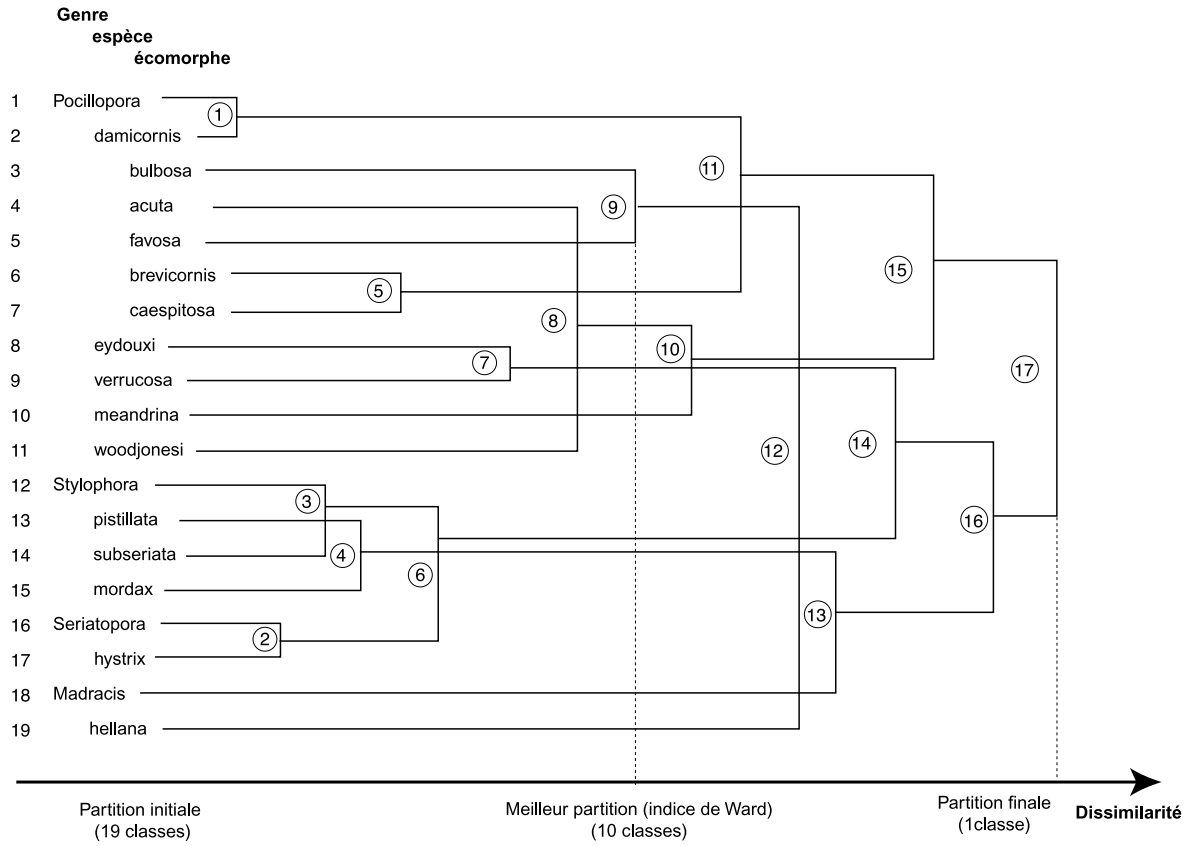


FIG. 5.3 – *Dendrogramme obtenue par Classification Ascendante Hiérarchique des classes de la famille des Pocilloporidae.*

la cohérence de l'ensemble. Dans l'exemple précédent, la méthode associe en premier lieu les classes *Pocillopora* (genre) et *Pocillopora damicornis* (espèce), puis *Seriatopora* (genre) et textitSeriatopora hystrix (espèce) à l'étape 2, ce qui est cohérent d'un point de vue Systématique. Par contre, nous pouvons nous demander pourquoi à l'étape 12, la classe *acute* a-t-elle été associée à la classe *hellana* ? Cela a-t-il un sens quelconque ? Il faut cependant considérer que au plus le nombre de classes diminue au plus des classes hétérogènes sont associées et les résultats deviennent peu signifiants. On peut considérer dans l'exemple précédent qu'au delà de l'étape 9 (étape indiquée par l'indice de Ward), les classes obtenues ne sont plus signifiantes.

5.6.4 Identification polythétique

La mesure de dissimilarité permet de situer une description ω non classifié par rapport à un ensemble Ω de descriptions de référence. Plusieurs stratégies sont alors applicables dans le but de déterminer la classe de la description considérée :

– la classe de l'individu le plus proche (le plus proche voisin) est affectée à celle de l'individu ω à déterminer. La méthode consiste à comparer tout individu de Ω à ω, à l'aide de la mesure de dissimilarité.

– la classe la plus fréquente dans la liste ordonnée $Y \subseteq \Omega$ formée par les k individus les plus proches de ω. Cette stratégie est du type k plus proches voisins.
– utiliser Y comme ensemble d'apprentissage pour la construction de l'arbre d'identification exposé au chapitre précédent. L'identification est alors réalisée en deux étapes. Dans la première étape, l'utilisateur renseigne un certain nombre de descripteurs selon son libre choix, puis à l'aide de l'arbre d'identification obtenu à partir de l'ensemble d'apprentissage Y détermine la classe associée par un parcours de l'arbre (seconde étape).

5.6.5 Expérimentation : Identification polythétique par différentes mesures de distance

Le tableau suivant illustre les résultats obtenus par la méthode d'identification *polythétique* de type k plus proches voisins exposée ci-dessus.

Différentes mesures de distances sur des données hétérogènes ont été utilisées. La distance euclidienne usuelle, les différentes mesures utilisées en Apprentissage Automatique que nous avons mentionnées précédemment (HOEM, HVDM, IVDM, WVDM) et la mesure globale que nous proposons (notée **DGR** pour Diatta-Grosser-Ralambondrainy).

Les résultats sont calculés selon le principe de validation croisée exposé au chapitre précédent (chap. 4). Les données considérées sont également les mêmes. Notons que dans ce contexte, seule la partie locale de la mesure DGR est appliquée, étant donné que les structures ne sont pas considérées.

Résultats

Ces résultats illustrent une augmentation très significative du pourcentage d'identifications correctes par la méthode *polythétique*, par rapport à la méthode *monothétique*.

En effet, pour la méthode *monothétique*, la moyenne obtenue sur l'ensemble des bases est de **70,40%** (cf. 4.1). Dans le cas présent, elle est de **80,48%**, soit une augmentation de **10%**.

Cette augmentation s'explique simplement par le fait que la totalité des descripteurs est utilisée par la méthode polythétique, alors que l'approche monothétique ne considère qu'un sous-ensemble de descripteurs correspondant à un chemin unique de l'arbre d'identification. La méthode est donc plus tolérante aux erreurs de descriptions, à l'imprécision et aux données manquantes. De plus, elle correspond davantage au *raisonnement synthétique* que font les experts pour situer globalement un objet dans une hiérarchie de taxons (cf. chap. 4).

5.6.6 Expérimentation 2 : Identification polythétique avec la mesure globale

Dans cette expérimentation, la mesure DGR s'applique dans sa partie locale et globale. De la même manière que l'expérimentation 4.2, les résultats obtenus sont exprimés par un intervalle : la borne inférieure de l'intervalle correspond au pourcentage le plus pessimiste de bon classement, la borne supérieure, au plus optimiste.

Nous rappelons également les résultats de la méthode par arbre d'identification afin de faciliter la comparaison.

Données non-structurées	Euclid	HOEM	HVDM	IVDM	WVDM	**DGR**
Annealing	94,99%	94,61%	94,61%	96,11%	95,87%	98,87%
Audiology	60,50%	72,00%	77,50%	77,50%	77,50%	76,00%
Audiology test	41,67%	75,00%	78,33%	78,33%	78,33%	88,46%
Bridges	58,64%	53,73%	59,64%	60,55%	56,64%	60,19%
* *Pocilloporidae* (coraux)	51,12%	53,24%	59,60%	59,60%	59,60%	**61,06%**
* *Siderastreidae* (coraux)	72,80%	82,00%	85,16%	85,40%	85,40%	**86,80%**
Echocardiogram	94,82%	94,82%	94,82%	100,00%	100,00%	82,58%
Flag	48,95%	48,84%	55,82%	57,66%	58,74%	46,39%
Hepatitis	77,50%	77,50%	76,67%	82,58%	79,88%	78,71%
Images segmentation	92,86%	93,57%	92,86%	92,86%	93,33%	98,10%
LED+17 noise	42,90%	42,90%	60,70%	60,70%	60,70%	60,70%
Monks-1	77,58%	69,43%	68,09%	68,09%	68,09%	79,83%
Monk2-2	59,04%	54,65%	97,50%	97,50%	97,50%	96,50%
Monk2-3	87,26%	78,49%	100,00%	100,00%	100,00%	100,00%
Mushroom	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%
Soybean (large)	87,26%	89,20%	90,88%	92,18%	92,18%	89,58%
Soybean (small)	100,00%	100,00%	100,00%	100,00%	100,00%	100,00%
* *Hyalonema* (éponges)	49,21%	53,50%	55,12%	55,12%	55,12%	**56,80%**
Vehicle	70,93%	70,22%	70,93%	69,72%	65,37%	79,02%
Zoo	97,78%	94,44%	98,89%	98,89%	98,89%	99,11%
Moyenne	71,64%	73,63%	79,31%	79,99%	79,57%	**80,%**

TAB. 5.3 – *Résultat d'identification polythétique par validation croisée, avec différentes mesures de distance, sans utilisation des structures.*

Interprétation

Ces résultats montrent de manière claire le gain obtenu par la méthode d'identification fondée sur la mesure de dissimilarité, par rapport à l'approche monothétique. Nous avons calculé en moyenne **5%** d'augmentation sur une analyse plus large. Ces résultats sont tout à fait significatifs dans le contexte de descriptions complexes issues de l'observation, de l'interprétation et de la description de spécimens réels.

Notons que ces résultats reflètent également la *qualité* et la *robustesse* des bases de connaissances, que l'on peut interpréter d'une manière générale par les facteurs suivants :

1. Qualité des modèles descriptifs. Choix des composants, des attributs, structuration.
2. Richesse des illustrations et des annotations.
3. Qualité des descriptions. Nombre de valeurs manquantes et inconnues, valeurs erronées.
4. Exhaustivité des bases, représentativité des différents taxons.

Données structurées	Monothétique	DGR
* *Pocilloporidae* (coraux)	[63% 79%]	[66,5% 84,7%]
* *Siderastreidae* (coraux)	[80% 84%]	[90,1% 92,4%]
* *Hyalonema* (éponges)	[59% 75%]	[61,8% 78,9%]

TAB. 5.4 – *Résultat d'identification par la méthode polythétique avec la mesure DGR globale.*

5.7 Conclusion

Nous avons proposé une mesure de dissimilarité entre objets structurés décrits par de valeurs complexes qui prend en compte la structure de ces objets, ainsi que la structure des descripteurs hiérarchiques. Cette mesure est utilisée par un ensemble de méthodes de classification et d'identification développées au sein de la plate-forme $\mathcal{IKBS}$:

1. Classification Ascendante Hiérarchique.
2. Classification par méthodes des nuées dynamiques distribuées.
3. Classification conceptuelle.
4. Identification *polythétique*.

D'autre part, les différentes expérimentations (classification et identification) visent à montrer l'apport de la mesure générale proposée par rapport aux mesures plus classiques qui ne prennent pas en compte toute la complexité des descriptions d'objets biologiques.

En particulier, nous montrons l'intérêt de la méthode de classification CAH pour la Systématique, ainsi que l'avantage de la mesure *polythétique* par rapport à la méthode *monothétique* par arbre d'identification exposée au chapitre précédent. D'autre part nous mettons en avant l'intérêt d'utiliser la structure des descriptions pour augmenter les performances des résultats d'identification.

Quatrième partie

La plate-forme à objets $\mathcal{IKBS}$

Chapitre 6

La plate-forme à objets $\mathcal{IKBS}$

Sommaire

6.1 Introduction

$\mathcal{IKBS}$ [100] est la plate-forme logicielle que nous avons développée pendant cette thèse afin de répondre à la nécessité, pour les biologistes, de disposer d'outils informatiques mettant en oeuvre

100. *Iterative Knowledge Base System ou Système itératif de gestion de Bases de Connaissances.*

la méthode expérimentale ("conjecturer et tester") de nature inductive (Grosser & Conruyt 1999, Conruyt & grosser 1999, Conruyt et al. 1998). La plate-forme $\mathcal{IKBS}$ est développée dans le langage Java selon le paradigme *orienté-objet* de conception et de développement d'applications.

Ce chapitre montre différents aspects de la plate-forme et de son utilisation. Plusieurs aspects sont identifiés et exposés progressivement du plus haut (utilisateur) jusqu'au plus bas niveau (programmeur) :

- Aspects méthodologiques ;
- Niveau utilisateur, interface et langage ;
- Architecture générale de la plate-forme.

6.2 Aspects méthodologiques de gestion des connaissances

Le processus de conception d'une base de connaissances (BC) est long et nécessite d'adopter une méthodologie précise fondée sur la formalisation, le traitement et la validation de connaissances produites. Plusieurs méthodologies générales pour l'*Ingénierie des connaissances ou Knowledge Engineering* semblent avoir du succès actuellement, dont CommonKADS (Shreiber et al. 1994), PROTÉGÉ-II (Puerta et al. 1992) et MKSM sont quelques exemples représentatifs. Ces méthodes d'acquisition sont bien adaptée à la résolution de problèmes industriels : CommonKADS par exemple, structure les connaissances à différents niveaux : stratégie, tâche, inférence, domaine, et utilise l'inférence déductive pour réaliser ces objectifs (Wielenga et al. 1992). Ces méthodologie sont bien utiles au développement de systèmes experts pour résoudre des problèmes complexes, effectuer des diagnostics ou anticiper des pannes de systèmes industriels sophistiqués.

En Sciences de la Vie, les experts appliquent une méthode expérimentale de nature essentiellement *inductive*, fondée sur *observation des faits, la constitution d'hypothèses, et des tests expérimentaux pour les mettre à l'épreuve* (Conruyt 1994). Les connaissances sont fréquemment remises en cause, modifiées, la terminologie évolue. Il est donc essentiel que la méthode de gestion de connaissances utilisée reflète les habitudes des experts, afin de faciliter le processus d'acquisition à la fois des connaissances et des données du domaine traité.

Le problème majeur qui se pose dans ce contexte d'évolution rapide des données/connaissances est celui de la mise à jour et la cohérence globale de la BC, en particulier les relations qui lient les instances aux concepts. En effet, comme nous l'avons développé (chap. 2), les instances sont attachés à un concept (modèle descriptif) qui lui donne sa structure et l'ensemble des éléments pour le décrire (descripteurs). Ainsi, lorsqu'un modèle est remis en cause, comment faire en sorte que l'ensemble des descriptions reste cohérent ?

La méthodologie de gestion des connaissances que nous proposons met en oeuvre une démarche **itérative** (ou incrémentale) de gestion des connaissances. Elle s'inspire de la spirale de Boehm (1988), élaborée pour la conception d'applications (*software engineering*).

La construction d'une base de connaissances se fait par un processus d'aller-retour entre les connaissances formalisées, les données contextuelles et les résultats produits par les méthodes d'analyse. Les outils interactifs d'acquisition d'$\mathcal{IKBS}$ offrent la possibilité de remettre fréquemment en cause le travail préalablement effectué.

6.2.1 Deux types de connaissances

Deux types de connaissances sont distingués :

1. Les connaissances en Systématique. Elles correspondent à une modélisation des connaissances des experts concernant la Taxonomie de espèces et la description de spécimens réels en collection. Elles intègrent également les connaissances de type savoir-faire (connaissances tacites) à travers la structuration naturelle du domaine, les règles mises en jeu et les poids associés aux descripteurs qui caractérisent les descriptions.
2. Les données ou informations contextuelles. Elles concernent tout ce qui a trait au domaine et que les experts jugent pertinent d'intégrer à la BC. Elles ne sont cependant pas modélisées explicitement à l'aide de l'outil informatique, mais référencées depuis les éléments présents dans la BC. Ces données peuvent ultérieurement être modélisées et intégrées à la BC. Les experts disposent en effet d'un ensemble d'éléments, qui sont des références à des travaux et ouvrages existants, des monographies, des données photographiques, géographiques, etc. Ces informations alimentent la réflexion de l'expert en apportant une quantité de renseignements pour les connaissances à modéliser. Les nouvelles technologies hypermédias et hypertextuelles offrent en particulier un support idéal pour organiser ces différents types d'information. Les données contextuelles peuvent également être construites spécifiquement par les experts avec l'aide des informaticiens pour les besoins de la BC, dans le but d'améliorer globalement l'*intelligibité* des bases.

Pour résumer, nous pouvons considérer que les BCs construites sont constituées par un "noyau d'expertise" (connaissances formalisées), en relation avec un ensemble d'informations de différentes natures, relatives au domaine (connaissances contextuelles).

6.2.2 L'itération

Dans le but de mettre en œuvre la méthode scientifique en biologie (conjecturer et tester) (Popper 1973), notre méthodologie suit le processus naturel d'apprentissage des connaissances par un expert. Ce processus est *itératif* dans la mesure où il procède par aller-retour entre des phases d'acquisition des connaissances, de traitement et de validation des connaissances produites :

- Acquisition des connaissances. Acquisition des modèles descriptifs, des descriptions, des données contextuelles, etc.
- Traitement des connaissances. Par les méthodes d'identification et de classification exposées dans les chapitres précédents.
- Validation et remise en cause (itération) des connaissances modélisées, mais également des données contextuelles.

La figure 6.1 illustre le cycle de gestion *itérative* des connaissances.

Acquisition des connaissances

Lors de la phase d'acquisition des connaissances, nous distinguons d'une part l'étape d'acquisition du modèle descriptif (l'Observable) et d'autre part l'étape d'acquisition des descriptions

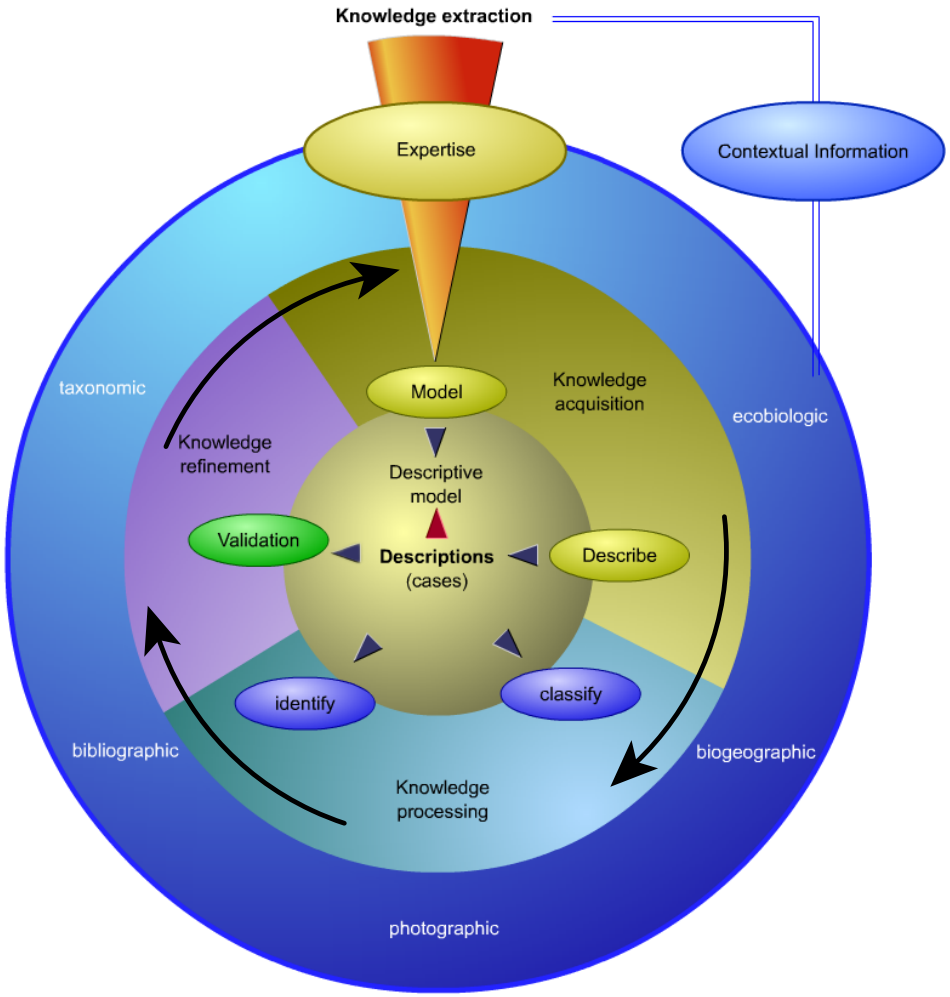


FIG. 6.1 – *Le processus* ***itératif*** *de gestion des connaissances.*

(l'Observé) et d'autre part l'étape d'acquisition des données contextuelles qui sont indexées aux connaissances modélisées.

Un modèle descriptif complet, exhaustif et non ambigüe, étant très difficile à construire *a priori*, un premier modèle simple est élaboré. Ce modèle représente un concept du domaine, généralement un taxon donné (Genre ou Famille). Dans un deuxième temps, un ensemble de descriptions est effectué, sur la base d'un ensemble de spécimens issus des collections. Nous préconisons de décrire un cas par espèce présente dans le taxon.

Les étapes suivantes offrent la possibilité de remettre en cause le modèle initial, les outils informatiques mettant à jour de manière automatique les descriptions renseignées au préalable. Il est très important de ne pas remettre en cause l'ensemble des descriptions effectuées, à chaque modification d'un modèle. La phase d'acquisition des connaissances est en effet de loin la plus couteuse en temps de travail pour l'expert et il n'est pas rare qu'il faille au moins vingt aller-retours entre la définition d'un modèle descriptif et la constitution d'une base de cas stabilisée.

Les modifications du modèle descriptif peuvent concerner par exemple :

- L'ajout ou la suppression d'un composant ;
- La modification du statut d'un composant (fictivité, absence possible, etc.) ;
- L'ajout ou la suppression d'un attribut ;
- Le changement du type d'un attribut. En particulier du type taxonomique vers le type nominal, ou inversement. Les valeurs des cas sont adaptées automatiquement ;
- L'ajout ou la modifications de règles, d'informations contextuelles, ou de commentaires à un élément descriptif existant ;
- Le déplacement d'un attribut vers un autre composant ;
- la création, la modification ou la suppression des relations ;
- La modification du domaine de valeurs d'un attribut ;
- etc.

Traitement des connaissances

Les méthodes de traitement automatique des connaissances que nous avons développées aux chapitres 3, 4 et 5, permettent de tester la qualité des BCs :

- Opérations internes du modèle CoDesc (chap. 3). La construction automatique de descriptions généralisantes et de définitions de concepts offre une vision synthétique des représentations présentes dans la BC.
- Identification monothétique automatique (chap. 4). Le mécanisme de validation croisée permet de déterminer les pourcentages de bons classements des descriptions (de spécimens ou synthétiques), qui reflètent la qualité de la BC par rapport à un ensemble restreint de descripteurs, qui sont les plus discriminants.
- Méthodes d'analyse fondées sur la comparaison des descriptions (chap. 5). Les méthodes de classification Ascendante Hiérarchique, d'identification polythétique, de classification conceptuelle offrent aux experts un ensemble d'outils permettant d'explorer les connaissances. L'outil d'identification automatique par validation croisée permet de mettre en évidence la qualité des BCs sur la base de la totalité des descripteurs présents dans les descriptions.

Validation

Cette phase se fait en permanence tout au long du processus de construction de la BC, à la fois au cours des phases d'acquisition et de traitement des connaissances. Pour la phase de traitement, les différentes méthodes mettent en évidence les anomalies, les erreurs de description ou les incohérences du modèle. Les données contextuelles interviennent pendant la phase de validation : elles sont considérées comme une vision extérieure, des connaissances telles qu'elles sont admises par la communauté scientifique du domaine.

Les mécanismes de gestion de l'évolution des connaissances mis en oeuvre offrent la possibilité de valider ou de remettre en cause les connaissances présentes dans la BC.

6.3 Niveaux de représentation des connaissances dans $\mathcal{IKBS}$

Afin de mettre en oeuvre la méthodologie itérative, différents outils ont été développés. Avant de présenter ces outils, nous introduisons les différents niveaux de représentations des données et connaissances, du point de vue de l'utilisateur, ainsi que du point de vue de la plate-forme informatique, afin de situer globalement les différents niveaux d'action possible selon que l'on souhaite utiliser, ou bien faire évoluer, la plate-forme $\mathcal{IKBS}$.

6.3.1 Trois niveaux de représentation

D'un point de vue informatique, nous pouvons distinguer trois niveaux de représentation des données et des connaissances manipulées par $\mathcal{IKBS}$.

1. Le **méta-niveau** ou niveau-classe : implantation du modèle de représentation CoDesc , à l'aide de classes du langage de programmation orienté-objet Java (LPOO).
2. Le niveau **Codesc** ou niveau objet : les objets (ou instances des classes) du LPOO constituent le modèle interne des connaissances du domaine.
3. Le niveau **interactif** : les instances sont manipulées par l'utilisateur à l'aide du *langage de définition* ou de l'*interface graphique*.

6.3.2 Le méta-niveau

Le modèle CoDesc est implanté par l'intermédiaire de classes du LPOO. Ces classes définissent un ensemble de structures et de types abstraits de données, adaptés à l'expression des différents éléments constitutifs du modèle : composants, attributs, règles, domaines, valeurs, etc. Toute modification de la sémantique de CoDesc se situe à ce niveau et n'est possible qu'à travers la programmation. Ce niveau n'est donc accessible qu'aux programmeurs.

Une étude de ce niveau sera développée au § 6.5.

6.3.3 Le niveau CoDesc

Toute entité active[101] du modèle CoDesc : modèles descriptifs, composants, cas, valeurs, relation, règles, etc. est une instance d'une classe du méta-niveau. Ces objets ne sont pas isolés, dans la mesure où ils partagent des relations avec les autres objets. L'ensemble des objets est organisé en une structure de graphe, appelé *graphe d'accointance*. L'état de chaque objet et la topologie du graphe d'accointance reflète l'état d'une base de connaissances à un instant donné. Notons que les objets du LPOO ne sont ni visibles ni intelligibles directement par l'utilisateur. Ils le sont par l'intermédiaire de différentes vues et éditeurs graphiques (niveau interactif), sous forme textuelle (langage CoDesc) ou bien par l'intermédiaire de pages HTML générées.

101. Par active, nous entendons présente en mémoire à un instant donné.

Connexité du graphe d'accointance

Pour un domaine d'application donné, le graphe d'accointance est *connexe* : aucune entité n'est isolée du reste du graphe. La connexité est garantie par :

1. La **Hiérarchie** des concepts. Tout concept $\mathcal{C}$ (dénoté par un modèle descriptif) est situé dans la hiérarchie des concepts $\mathcal{H}$, de racine unique le concept $\mathcal{C}_0$ (cf. chap. 2). Les différents concepts d'un domaine d'application sont donc liés par des relations de spécialisation. Ces relations sont réifiées en instances d'une classe du méta-modèle (`RelationSpec`) dénotant la relation de spécialisation. La relation de spécialisation du niveau conceptuel (CoDesc) est ainsi un objet du LPOO.
2. Les relations de **Composition**. Tout les composants d'un modèle descriptif sont liés par des relations de composition (comme nous l'avons vu au niveau conceptuel), réifiées en instances d'une classe (`RelationCompo` dénotant la relation de composition. Le modèle descriptif possède une racine unique de type *Schéma*. Cet objet Schéma est donc un composant particulier du modèle, mais il désigne également le concept dans son intégralité, dans le sens où tout élément descriptif est accessible à partir de la donnée d'un Schéma. Ceci se fait de deux manières : soit par un parcours récursif en profondeur du modèle descriptif (structure d'arbre), soit par accès direct par l'intermédiaire du nom d'identificateur (unique) de l'entité recherchée. De même, tout attribut, règle, commentaire, etc. est attaché à un composant du modèle descriptif et donc accessible par l'intermédiaire de son composant.
3. **Lien "est-un"**. Chaque cas est lié à un modèle descriptif par une relation de type "est-un". Cette relation est également réifiée en instance d'une classe (`RelationIsa`). Les cas sont regroupés pour former une "base de cas" par l'intermédiaire d'une structure qui les contient (classe `SchemaValue`). Différentes bases de cas peuvent ainsi être gérées par cet objet de plus haut niveau.
4. **Valeurs**. Les valeurs complexes sont également réifiées en instances de la hiérarchie `Value` qui sont liées d'une part au cas qu'elles renseignent et d'autre part à l'attribut ou au composant du modèle descriptif par un lien appelé *type*.

Usage de l'introspection

L'utilisateur peut avoir connaissance de l'état des objets du graphe d'accointance par l'intermédiaire de *fonctions introspectives*. Java offre en effet des mécanismes d'**introspection** qui permettent par exemple, lors de l'exécution de l'application, d'accéder à la classe d'un objet, de recueillir la liste des méthodes et des propriétés associées à cette classe afin de déclencher un envoi de message explicite à l'objet. Les fonctions introspectives sont particulièrement utiles pour le développeur lors de la mise au point des programmes, car elles permettent d'accéder à l'état des objets sans recourir à une interface externe. Ainsi par exemple, les opérations formelles sur les valeurs complexes (cf. chap. 3) peuvent être exécutées directement dans la champ d'évaluation des valeurs. L'évaluation de la commande : `=union([5 10], [7 12])` retourne l'intervalle `[5 12]`.

6.3.4 Le niveau interactif

Toute définition et modification d'éléments du modèle peut être réalisée par une action graphique de l'utilisateur. Par exemple : définir un nouveau composant, un attribut, une règle, une relation, dupliquer un composant ou un sous-arbre, renseigner des valeurs, etc.. La métaphore retenue pour les manipulations graphiques est celle utilisée en particulier par les environnements de programmation orienté objets : l'éditeur de composant permet d'*inspecter* le contenu d'un composant : la liste de ses propriétés, l'ensemble des règles et des illustrations associées, etc..

Ces définitions peuvent également être effectuées par l'intermédiaire du **langage de définition CoDesc**, dont la syntaxe précise est donnée à l'annexe C.

Nous développerons à la section 6.4 de manière détaillée le niveau interactif à l'aide des outils graphiques d'$\mathcal{IKBS}$ et du langage de définition.

6.3.5 Utilisateur et système

Nous présentons la figure suivante (fig. 6.2), qui illustre les 3 niveaux exposés précédemment, tels qu'ils peuvent être perçus d'une part par l'utilisateur et d'autre part considérés par le système.

Le modèle CoDesc est implanté par le concepteur **(1)** par un ensemble de hiérarchies de classes **(2)** d'un langage de programmation orienté objet. L'instanciation du modèle dans un domaine particulier **(3)** est une base de connaissances, qui peut être visualisée graphiquement **(4)** et manipulée par l'utilisateur à l'aide d'actions (primitives) graphiques **(5)** ou bien par le langage CoDesc **(6)**.

Au niveau utilisateur

L'utilisateur peut agir au niveau du modèle interne par l'intermédiaire du langage, ce qui induit les modifications correspondantes au niveau graphique. Inversement, les actions graphiques peuvent provoquer des changements internes au niveau du modèle (objets du LPOO). Certaines actions ne concernent cependant que les paramètres d'affichage et ne provoque pas de modification du modèle interne. Par exemple : afficher/masquer un sous arbre ou des attributs, agrandir la vue, mettre en forme l'arbre descriptif (formattage), etc.

Différentes vues du modèle interne sont possibles et peuvent être présentes simultanément. Par exemple, une vue complète et détaillée de l'ensemble des caractéristiques du modèle et une vue simple où n'apparaissent que les caractères les plus importants. Par exemple, la visualisation des objets composites du modèles uniquement.

Une caractéristique très intéressante que nous avons développé autorise la présence simultanée de plusieurs vues en langues différentes. Le modèle interne est le même, mais les vues sont différentes. Ainsi, une action dans un des deux modèles provoquera la modification correspondante dans le second modèle, mais dans une autre langue.

Ces fonctions de *gestion multi-langues des connaissances* ne sont pas encore totalement opérationnelles dans $\mathcal{IKBS}$. Nous souhaiterions poursuivre ces développements dans le futur. En effet, implantées au sein d'une architecture client-serveur (modèle interne côté serveur, vues côté client), elles autoriseraient l'édition simultanée et synchronisée du même modèle interne, par dif-

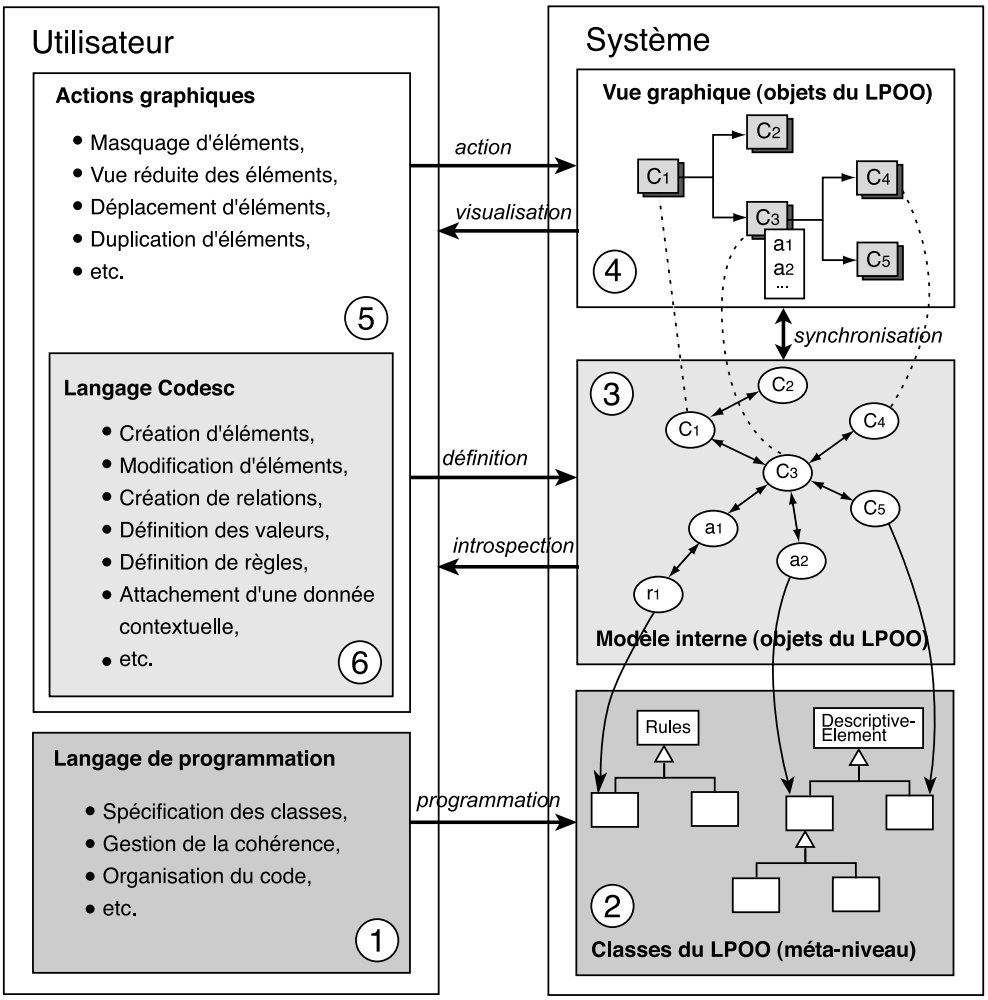


FIG. 6.2 – *Les 3 niveaux de représentation du modèle CoDesc, côté utilisateur et côté système.*

férentes personnes et en différentes langues. Cette perspective nous semble intéressante et adaptée à la problématique de la communauté des Systématiciens en ce qui concerne en particulier la définition des ontologies (terminologie consensuelle, thesaurus, etc.).

Au niveau système

Du point de vue interne à l'application, le principe de dissociation du modèle et des vues est inspiré du modèle MVC (Modèle-Vues-Contrôleurs) pour la construction d'interfaces graphiques. Dans ce modèle, les vues communiquent avec le modèle interne par l'intermédiaire de contrôleurs, ce qui permet au modèle d'être totalement indépendant des différentes vues, les contrôleurs se chargeant d'effectuer le lien et de propager les évènements de modification. Notre modèle est cependant une extension du modèle MVC, plutôt qu'une utilisation simple, nous en donnons ici seulement le principe général :

Les évènements sont réifiés en tant qu'objets du langage, instances d'une hiérarchie de classes traduisant des types d'évènements. La hiérarchie est organisée en fonction du type de l'élément

(composant, attribut, valeurs, etc.) sur lequel est survenue l'action. Les objets-évènements sont créés lorsqu'une action de modification survient sur le modèle interne. Ils encapsulent l'information concernant l'origine et l'émetteur de la modification. Dans un second temps, ils sont transmis via le contrôleur aux différentes vues du modèle interne, qui adaptent ses éléments de façon à prendre en compte les modifications.

6.4 Le niveau interactif

Nous développons maintenant de manière plus détaillée le *niveau interactif* permettant à un expert d'exprimer ses connaissances dans le formalisme CoDesc. Nous présentons successivement les outils qui permettent à un expert :

- De définir un modèle descriptif et ses constituants ;
- De renseigner une base de cas ;
- D'associer des données contextuelles aux éléments descriptifs.

6.4.1 Définition d'un composant

Un composant est décrit par une liste de *propriétés internes* et de *propriétés externes*. Les propriétés internes (fig. 6.3) définissent : son nom (label), son identificateur unique, son statut (fictif, absent possible, etc.) ainsi qu'un ensemble d'informations :

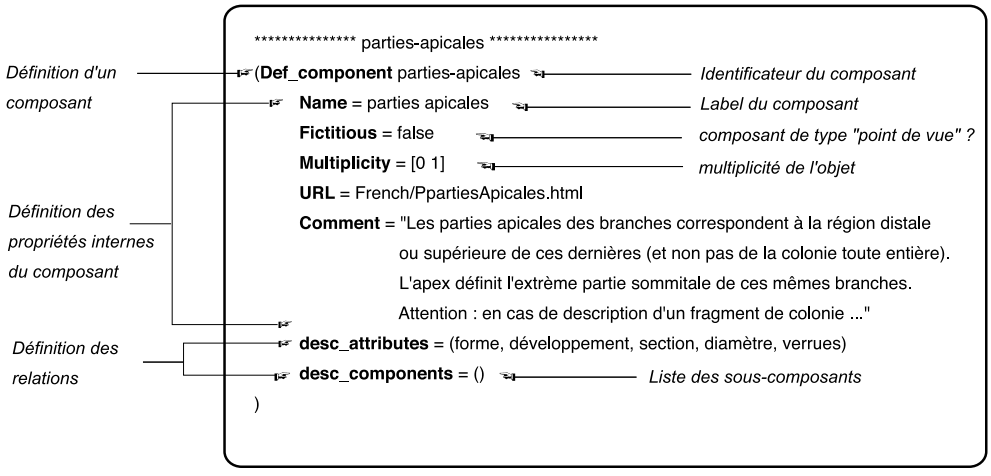


FIG. 6.3 – *Définition textuelle d'un composant. Un composant est défini par une liste de propriétés internes (identificateurs, label, commentaire, lien hypertexte, etc.), par une liste d'attributs et par une liste de sous-composants (pas de sous-composants présents sur cet exemple).*

- Un *commentaire* permettant de préciser la nature de l'entité considérée.
- Une liste d'*illustrations* définissant un ensemble de documents multimédia (textes, images, etc.) pouvant être attachés pour compléter sa définition.
- Une liste d'*URL* (adresses électroniques) permettant de relier l'entité à des données contextuelles accessibles sur le réseau.

Les propriétés externes définissent les relations vers l'ensemble des composants et d'attributs en relation, par l'intermédiaire de leurs identificateurs. Ces relations peuvent être de deux types : « sous-partie » (ou composition) et « sorte-de » (spécialisation ou subsomption). La définition complète d'un composant doit donc inclure la définition des attributs et les composants en relation.

Remarquons que contrairement aux modèles à objets, pour lesquels la définition des attributs est inclus dans la définition de la classe, la définition des attributs est extérieure à la définition des objets. La définition des attributs peut ainsi évoluer sans modification de la définition des composants.

En mode graphique, les composants sont créés par une action de la souris sur la fenêtre globale (modèle descriptif). Cette action génère un composant par défaut qui peut alors être défini précisement par l'intermédaire de la fenêtre d'édition locale, appelé éditeur ou inspecteur de composant.

La figure 6.4 illustre la vue d'édition graphique, équivalente à la définition textuelle.

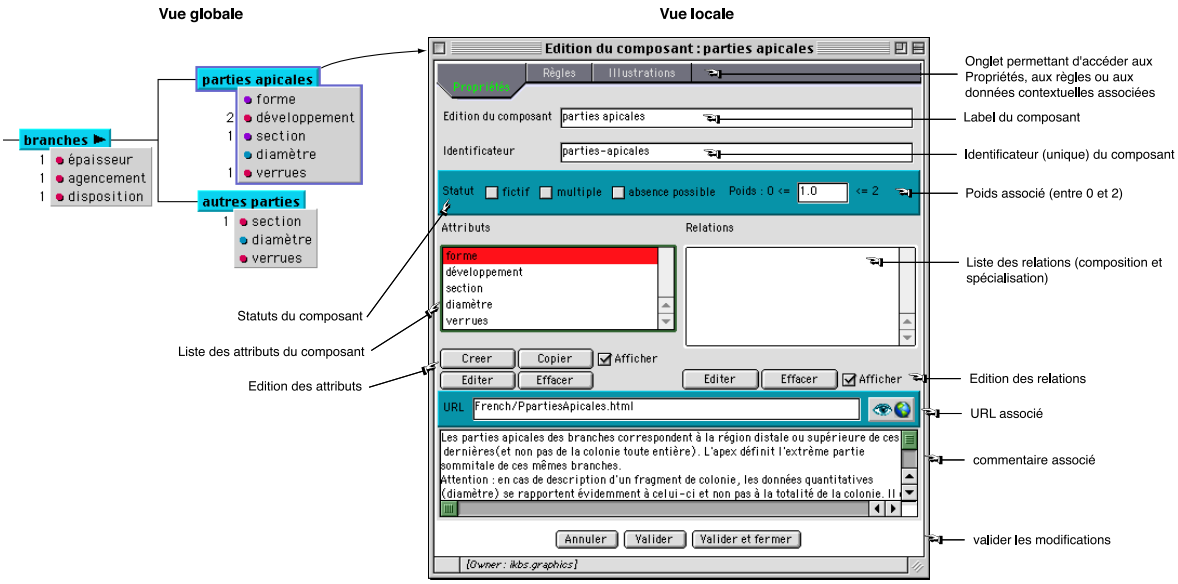


FIG. 6.4 – *Vue globale et vue locale du composant* ***Parties apicales*** *dans* $\mathcal{IKBS}$.

Différentes *facettes* d'édition sont accessibles lors de l'édition d'un composant :

- **Propriétés** : pour l'édition des propriétés internes et externes du composant,
- **Règles** : pour l'édition des règles associées,
- **Illustrations** : pour l'édition des photos et des données contextuelles associées au composant (cf. fig. 2.15).

6.4.2 Définition d'un schéma

Tout modèle descriptif possède nécessairement un composant de type Schéma, qui est la racine de l'arbre de description. Le Schéma est un composant particulier qui contient les même propriétés que les composants mais aussi un ensemble de propriétés internes spécifiques :

Propriétés spécifiques au Schéma

- Un *Auteur*, pour la gestion des droits de modification.
- Un *numéro de version*, pour la gestion des versions.
- Une *date de modification* pour la gestion des versions et des sauvegardes.

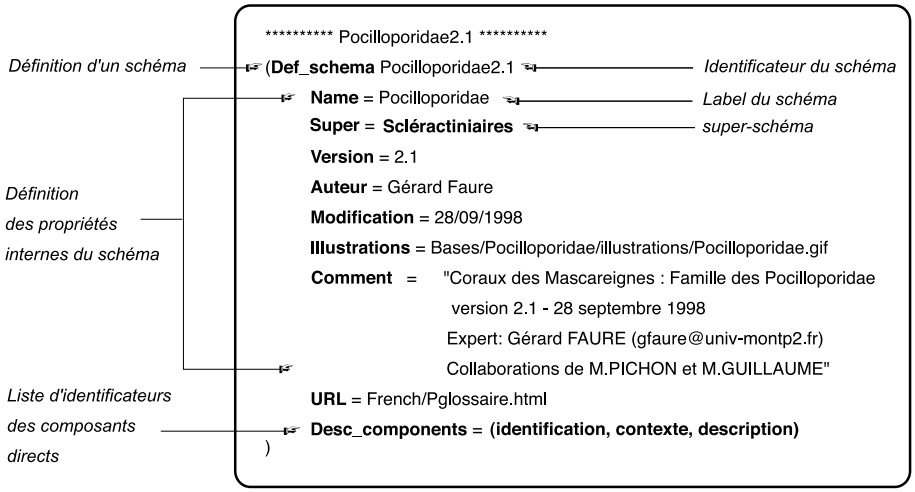


FIG. 6.5 – *Définition textuelle d'un schéma. Un schéma possède des propriétés internes spécifiques : version, auteur et date de modification.*

6.4.3 Définition d'un attribut

Un attribut représente une propriété (ou caractéristique) d'un composant. Tout attribut est attaché à un unique composant. Les propriétés associées à un attribut sont sensiblement équivalentes à celles qui définissent le composant et le schéma. En particulier, un atttribut dispose des trois facettes : Propriétés, Règles et Illustrations. Nous regroupons sous le terme générique *éléments descriptifs*, les composants, les attributs et les schémas.

L'identificateur d'un attribut (nom complet) est formé par la concaténation du label donné à l'attribut et de l'identificateur du composant auquel il est attaché. Cet identificateur est unique pour l'ensemble des attributs présents dans un modèle. L'identificateur est calculé automatiquement par le système, et ne peut être redéfini par l'utilisateur. Il peut donc changer au cours du cycle de vie du modèle : notamment lorsque son label ou celui du composant change ou lorsqu'il est déplacé vers un autre composant. L'identificateur reste cependant visible à l'utilisateur, car c'est le seul moyen de l'identifier de manière unique.

Le type de l'attribut (symbolique, numérique, taxonomique et texte) peut être défini par l'intermédiaire de l'éditeur local (graphique). Ce dernier modifie son aspect en fonction du contexte, en particulier pour les propriétés dépendantes du type, ainsi que la partie concernant l'édition du domaine de définition des valeurs.

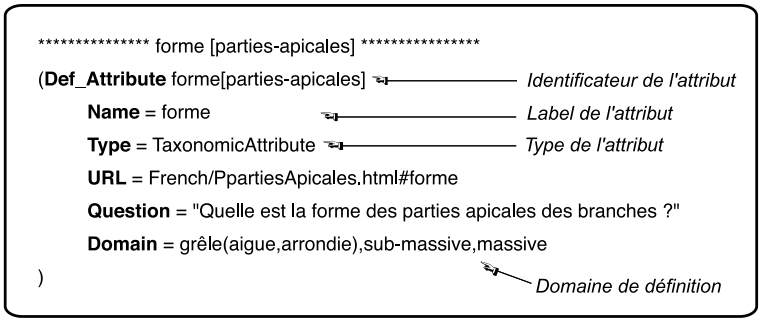


FIG. 6.6 – *Définition textuelle d'un attribut.*

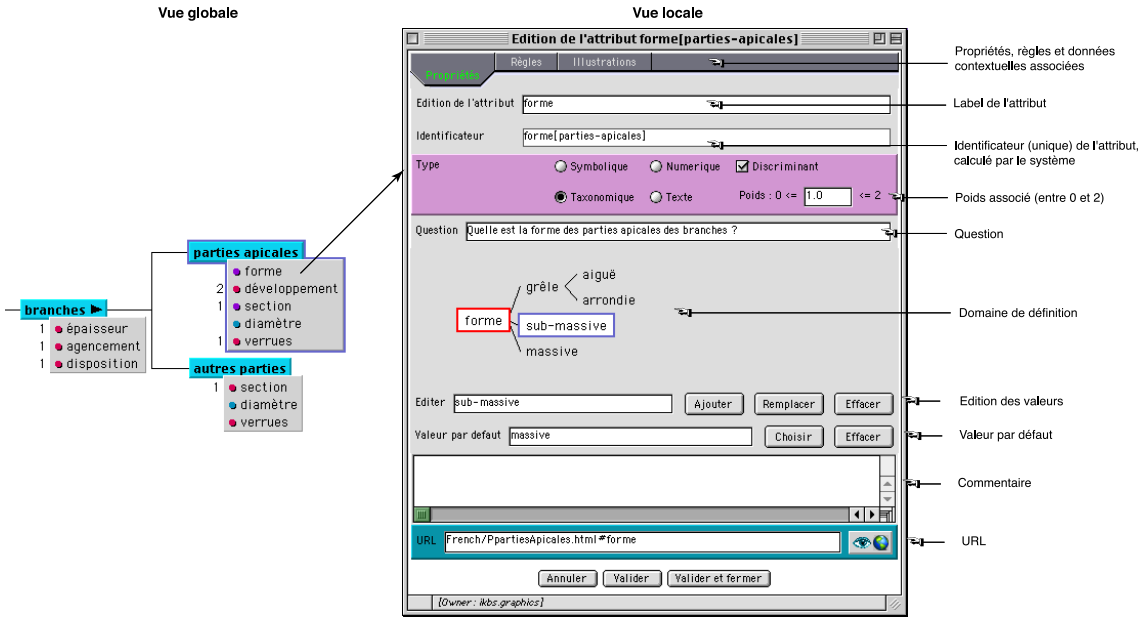


FIG. 6.7 – *Vues globale et locale de l'attribut* ***forme*** *du composant* ***parties apicales*** *de type taxonomique.*

6.4.4 Définition des règles

Des règles peuvent être associées à tout élément descriptif. La figure 6.8 illustre quelques règles associées à l'attribut **forme[parties-apicales]** :

```
Si taxon[identification] = Pocillopora damicornis ECO brevicornis ALORS sub-massive
Si taxon[identification] = Pocillopora meandrina ALORS massive
Si taxon[identification] = Stylophora mordax ALORS massive
```

Dans cet exemple, les règles définissent des contraintes entre l'attribut auquel est attachée la règle et la valeur associée à l'attribut de classe taxon.

Ces règles sont activées au niveau d'un cas, lorsque la valeur correspondant à la première partie (la prémisse) est renseignée. Celle-ci est constituée d'une assertion à valeur dans $\{vrai, faux\}$.

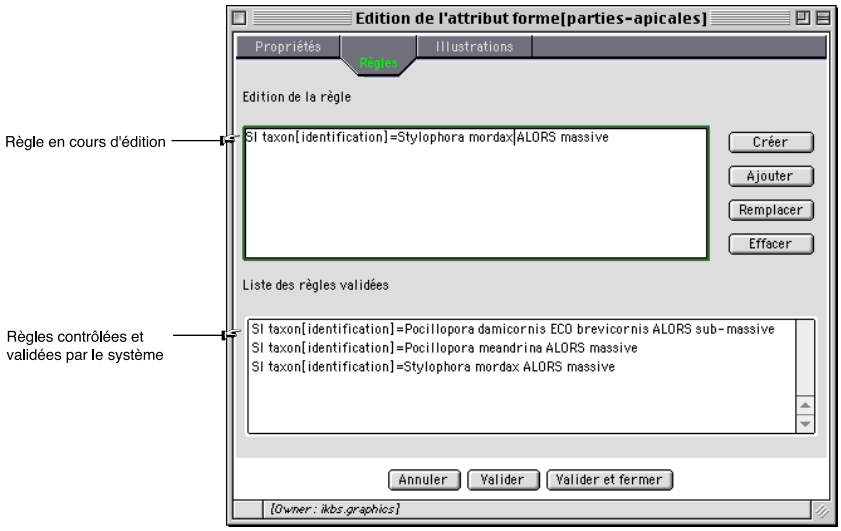


FIG. 6.8 – *Edition des règles de l'attribut* ***forme[parties-apicales]***

Notons que l'assertion constituant la première partie de la règle peut être une conjonction ou une disjonction. Les règles 2 et 3 peuvent se traduirent par :

```
Si taxon[identification] = Pocillopora meandrina
OU taxon[identification] = Stylophora mordax
ALORS massive
```

Pour la définition des assertions portant sur des attributs numériques, les opérateurs suivants peuvent être utilisés : $=$, $<$, $>$, $\geq$, $\leq$, pour exprimer des règles comme :

```
Si taille[calices] < 1.5 mm
ET diamètre[autres-parties] < 4 mm
ALORS taxon[identification] = Seriatopora hystrix
```

6.4.5 Outils pour l'acquisition des cas

L'acquisition d'une description peut être effectuée de deux manières, soit à l'aide de l'éditeur de cas d'$\mathcal{IKBS}$, soit par importation de tableaux de données.

Editeur de cas

La figure 6.9 illustre la fenêtre d'acquisition d'un cas. Lorsqu'un élément descriptif est sélectionné (composant ou attribut), l'environnement d'édition s'adapte au type de l'élément. Une

valeur peut être renseignée de plusieurs manières :

- Sélection d'un élément dans la liste. Pour les types qualitatifs, le domaine de définition est représenté par une liste de valeurs de type simple, celles du domaine de définition de l'attribut. Pour les composants, deux valeurs peuvent être sélectionnées : *absent* ou *présent*.
- Saisie dans le champs valeur. Pour le type numérique, les valeurs sont saisies dans un champ valeur, puis évaluées par le système. Ce champ est également utilisé pour la saisie de valeurs symboliques particulières : conjonctives, disjonctives ou exceptionnelles. La valeur *inconnu* peut être affectée à tout type d'attributs, ainsi qu'aux composants (absence ou présence inconnue).
- Résultat d'un *calcul*. Une valeur résultant d'un calcul peut être appliquée à un attribut. Nous développons ci-dessous la manière d'effectuer ces opérations.
- Résultat de l'activation d'une règle. Une valeur affectée à un attribut peut activer des règles (prémisse de la règle), provoquant l'affectation des valeurs calculées (partie conclusion) à d'autres descripteurs.

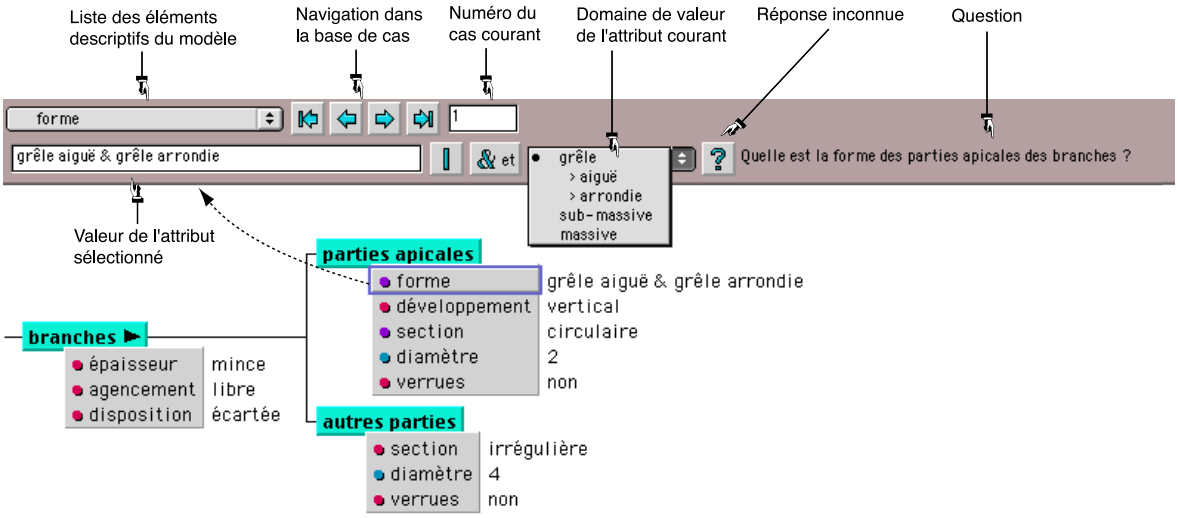


FIG. 6.9 – *Acquisition d'un cas (vue partielle du cas).*

Importation d'un tableau de données

Les cas peuvent être importés à partir de tableaux de données classiques ou de données mixtes (symboliques et numériques), de manière automatique. Un modèle descriptif sans relations, possédant un unique composant (Schéma) est créé. Il peut être transformé par l'utilisateur pour obtenir un modèle structuré. Les cas subissent de manière automatique ces transformations.

La figure 6.10 illustre le mécanisme de structuration. Deux étapes sont considérées : acquisition du tableau et structuration. Remarquons que les attributs qui possèdent des valeurs *non-applicables* (notées N-A) dénotent des propriétés contingentes (cf. chap. 2), c'est la raison pour laquelle elles sont attachées à un composant *absent-possible*.

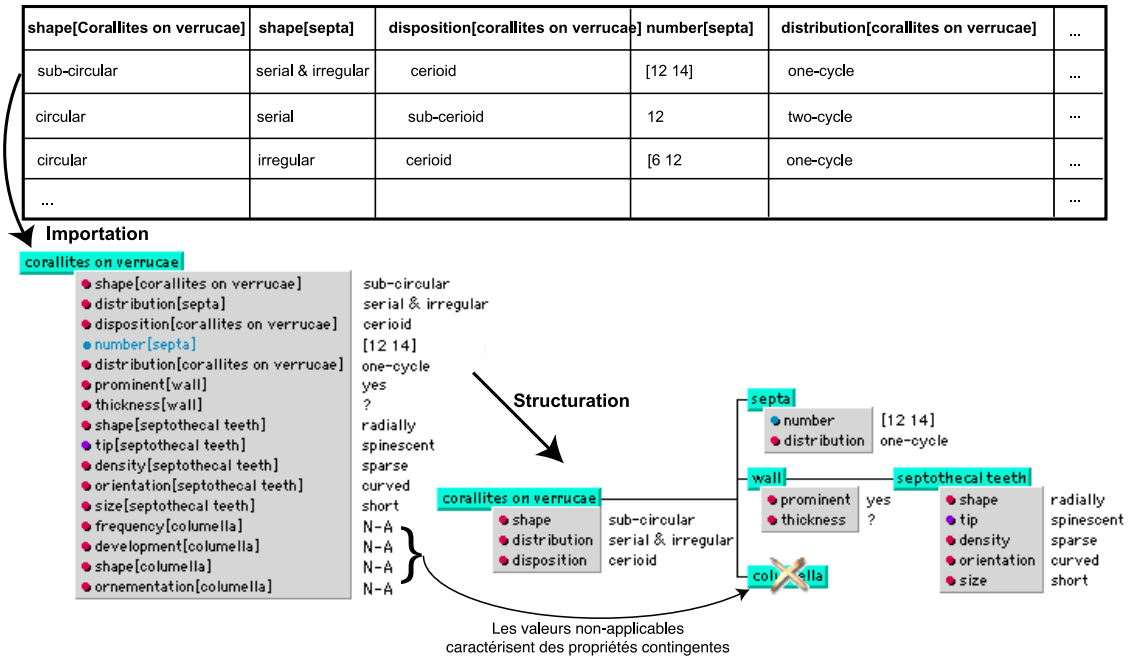


FIG. 6.10 – *Importation et structuration d'un tableau de données.*

Nous avons utilisé ce mécanisme pour importer les bases de l'*UCI Machine Learning Repository* afin de conduire les tests comparatifs effectués au chapitres 4 et 5.

6.4.6 Outil pour la gestion des données contextuelles

L'utilisateur dispose de la possibilité d'associer des données contextuelles de différents types aux composants et attributs du modèle, ainsi qu'aux valeurs des cas. L'interface proposée (fig. 6.11) dispose de puissantes fonctionnalités pour gérer, visualiser ou rechercher par mots-clefs ces données référencées dans la base.

Les données sont regroupées au sein d'une structure d'indexation, sorte de dossier virtuel qui les référence, appelé *fichier d'indexation*.

Types de données contextuelles

Les différents types de données contextuelles sont gérés de la manière suivante :

- Les données de type images (Jpeg, Gif, Tiff, etc.) peuvent être affichées directement avec l'outil, qui dispose de fonctions de zoom, de défilement automatique (SlideShow).
- Dossiers et fichier d'indexation. Ils définissent des données contextuelles composites : les dossiers correspondent aux "dossiers physiques" du système d'exploitation, les fichiers d'indexation regroupent un ensemble de références à des données contextuelles, éventuellement des dossiers ou des fichiers d'indexation. Les fichiers d'indexation et les dossiers permettent ainsi de construire une structure arborescente. L'activation d'un dossier ou d'un fichier d'indexation affiche son contenu.

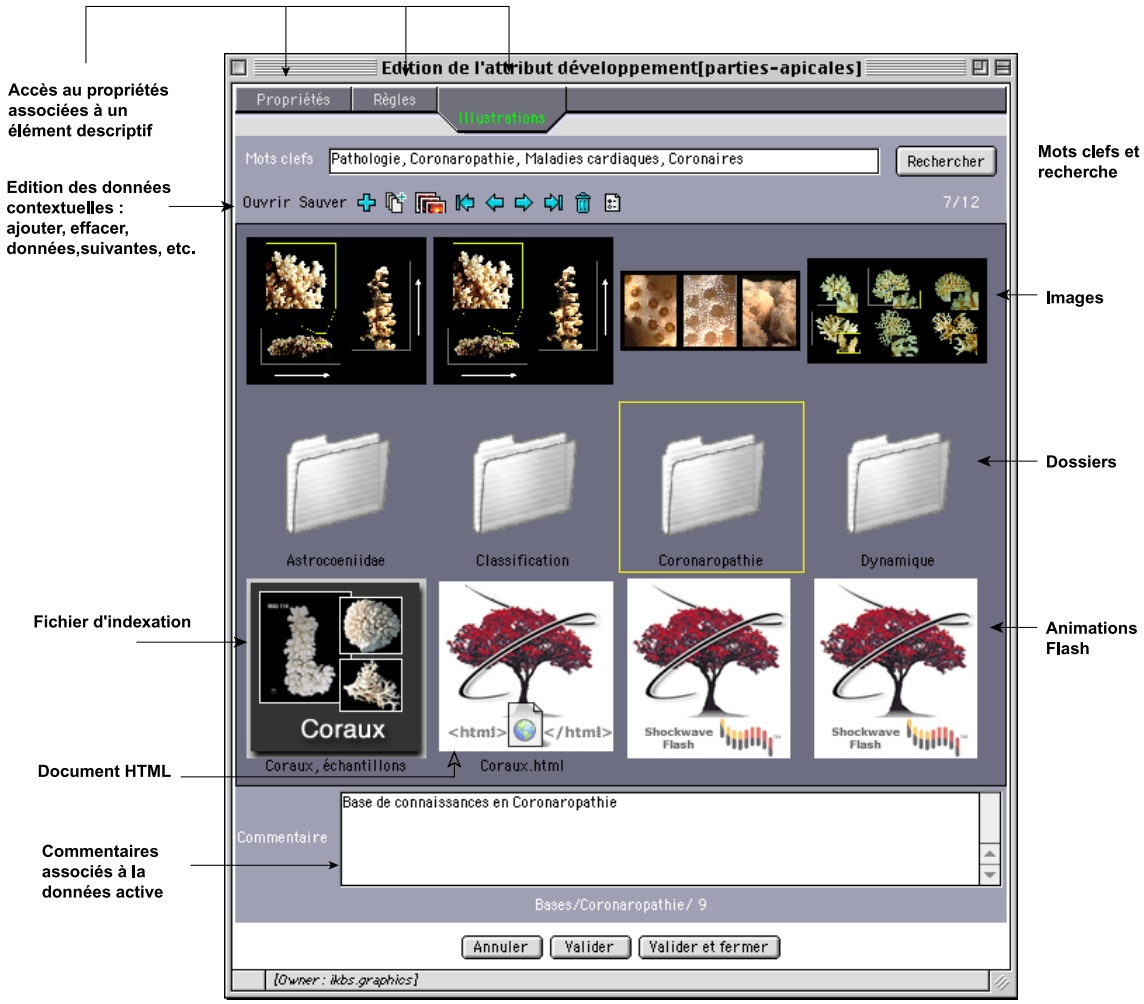


FIG. 6.11 – *Acquisition des données contextuelles.*

– Formats HTML, XML, Applets Java, Animations Flash, URL. Affichage délégué au navigateur.

– Formats RTF, PDF, ASCII. Affichage délégué à une application de type traitement de texte.

– Tableaux de données. Affichage délégué à une application de type "tableur".

Annotations et recherche par mots clefs

A toute donnée contextuelle peuvent être associés des annotations sous forme textuelle, ainsi qu'un ensemble de mots clefs. Ceux-ci sont utilisés par la fonction de recherche dans l'ensemble des informations référencées dans la BC.

Graphe de navigation

Les données composites (dossiers et fichier d'indexation) permettent de structurer hiérarchiquement les informations contextuelles. L'ensemble des informations est ainsi une structure arborescente ou une structure de graphe, car les cycles sont autorisés (un fichier d'indexation peut se référencer lui-même). Les noeuds de cette arborescence sont les données composites, tout autre donnée constitue une feuille de la structure. L'outil permet de visualiser et de naviguer dans la structure globale.

Génération de pages HTML

Cet outil est intégré à l'application $\mathcal{IKBS}$, mais peut également être utilisé indépendemment sous forme d'applet (pour être exécuté via un navigateur). Il dispose également de la possibilité de générer un graphe de pages HTML isomorphe au graphe de navigation. Les annotations et les mots clefs sont accessibles via des menus contextuels activés par des fonctions JavaScript.

6.4.7 Paramétrage des méthodes d'analyse

Les méthodes d'analyse exposées au chapitres 4 et 5 sont paramétrées et exécutées à partir de l'interface illustrée fig. 6.12. Les résultats sont affichées dans des fenêtres graphiques externes (arbres, graphes, treillis, dendrogramme, etc.), ainsi que sous forme textuelle dans la fenêtre de paramétrage.

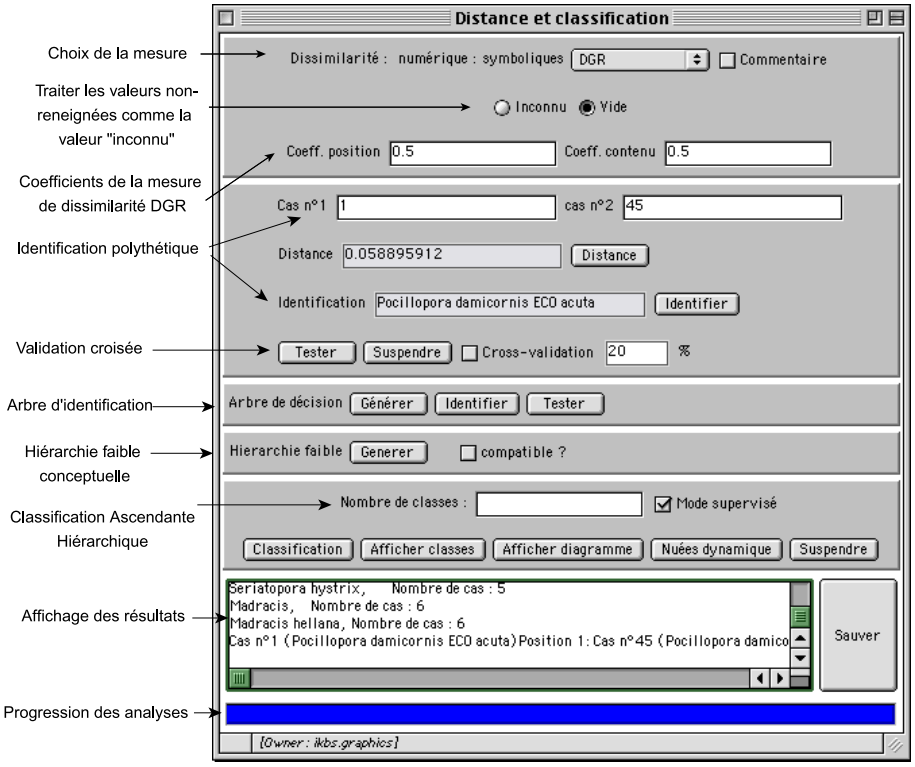


FIG. 6.12 – *Paramétrage des méthodes d'analyse.*

6.4.8 Conclusion et discussion

Nous avons exposé les différents outils informatiques mis à la disposition de l'expert pour l'acquisition, le traitement et la validation des connaissances, par l'intermédiaire du langage de définition CoDesc et des outils graphiques de la plate-forme $\mathcal{IKBS}$.

Ces outils ont été largement utilisés depuis plusieurs années pour la conception de la « base de connaissances sur les coraux des Mascareignes », ainsi que pour la construction de plusieurs prototypes de BC en Médecine, notamment dans le domaine du diagnostique de pathologies coronariennes *Coronaropathie*, en Systématique sur la famille des *Sertulariidae* (famille d'hydraires, petits organismes marins) et en Météorologie pour l'aide à la classification des cyclones. Il s'est avéré un outil puissant, interactif, convivial et simple d'utilisation pour la définition des concepts, la description des cas et les illustrations associées.

Nous abordons maintenant les aspects relatifs à la conception de la plate-forme et à son architecture logicielle.

6.5 Architecture générale de la plate-forme $\mathcal{IKBS}$

Afin de favoriser l'extensibilité et la réutilisabilité des différents composants logiciels, nous avons développé l'architecture d'$\mathcal{IKBS}$ selon le concept de « plate-forme à objets ».

6.5.1 Plate-forme à objets

La définition initiale décrit une plate-forme comme « un ensemble de classes qui représente une conception abstraite d'une solution pour une famille de problèmes, et qui permet la réutilisation à une granularité plus large que celui de la classe » (Johnson & Foote 1988). L'objectif d'une plate-forme est donc très ambitieux quant à ses possibilités de réutilisation et doit notamment permettre :

- la *réutilisation d'une architecture complexe* sous la forme d'un ensemble de classes abstraites définissant l'interface de flux et de contrôle de la plate-forme,
- la *réutilisation de composants de base* réalisant certaines parties de la plate-forme représentées par des classes concrètes d'objets.

La conception d'une plate-forme à objets est une des activités les plus délicates dans le domaine de la conception de logiciels à objets (Johnson & Foote 1988, Gamma et al. 1995). Trois types de logiciel à objets peuvent être identifiés : une application, une bibliothèque et une plate-forme à objets :

1. une **application** répond à des exigences limitées à un cadre bien précis. Elle n'offre pas de points d'ouverture et ne présente pas une architecture adaptée à la réutilisation.
2. une **bibliothèque** doit fournir un ensemble de composants réutilisables pour un domaine ou un type de traitement. La conception d'une bibliothèque est donc sensiblement plus complexe qu'une application.

3. une **plate-forme** doit non seulement offrir un certain nombre de composants de base comme le cas d'une bibliothèque, mais doit en plus définir une architecture réutilisable dans laquelle ces différents composants collaborent.

Dans ce cadre, il est alors difficile de garantir cette réutilisabilité *a priori*, c'est pourquoi le processus de conception d'une plate-forme est essentiellement **itératif**. Ces itérations mettent en jeu d'une part une phase de conception et de réalisation, et d'autre part une phase de réutilisation. Les échecs et les difficultés rencontrés lors de la réutilisation sont alors utilisés dans une nouvelle phase de conception pour améliorer la plate-forme. Selon la règle, en effet, un logiciel n'est réutilisable que s'il a été **concrètement réutilisé**. Une certaine stabilité peut ne jamais être atteinte à cause des évolutions des exigences fonctionnelles des différentes applications à réaliser, bien que les changements soient de moins en moins fréquents (Codenie et al. 1997). Une analyse du domaine visé et l'expérience issue de développements antérieurs d'applications sont toutefois très utiles pour commencer ces processus.

$\mathcal{IKBS}$ a été initialement développé pour la Systématique[102], mais des applications dans d'autres domaines sont actuellement en cours : en médecine nucléaire (*Coronaropathie*), en météorologie (classification des cyclones) et sur les plantes endémique de Maurice (projet Mauriflor@). La conception du système a subi des remises en cause profonde de son architecture à plusieurs reprises, afin de s'adapter d'une part aux évolutions des versions de Java et de répondre aux exigences des diverses applications. Le développement de la plate-forme a donc également suivi une démarche itérative dans la phase de conception, un aller-retour entre les fonctionnalités effectives et les besoins réels. A l'issue de ce cycle, la plate-forme s'est stabilisée (fig. 6.13).

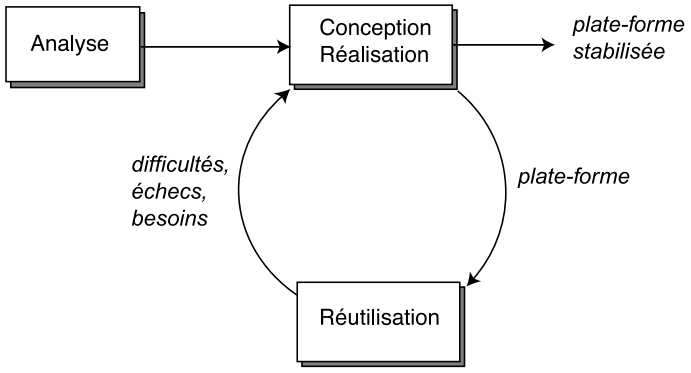


FIG. 6.13 – *Cycle de développement d'une plate-forme (Jaczynski 1998).*

6.5.2 Langage et architecture

$\mathcal{IKBS}$ est entièrement développé dans le langage orienté Java. Différents points d'entrées sont possible, selon que l'on souhaite utiliser "en-ligne" l'intégralité des fonctionnalités du système ou seulement certaines parties, comme l'identification par arbre ou l'édition de données contextuelles. En particulier, les arbres d'identification des familles des coraux sont accessible via le Web, indépendamment des outils de modélisation.

102. La Systématique des coraux (*Scléractiniaires*), ainsi que celle des Hydraires (*Sertulariidae*) et des éponges marines (*Hyalonema*).

Java

C'est un puissant langage orienté-objets qui intègre des notions avancées de programmation et de conception logicielle, telles que les notions d'*interface* (de communication), d'introspection, ou de réutilisabilité des composants logiciels (`JavaBeans`). Une bibliothèque très riche de classes, mises à jour régulièrement offre un ensemble de fonctionnalité très intéressantes, comme par exemple la communication avec les bases de données de tout type (`JDBC`), les objets distribués (`RMI`, `Servlets`), des structures de données évoluées (Dictionnaire, tables de hachage, collections, etc.) ou encore la possibilité de créer des applications embarquées (`Applets`) destinées à être exécutées par l'intermédiaire d'un navigateur.

Organisation des classes

Les classes standards sont organisées sous la forme de modules hiérarchisés, appelés *packages*, possédant une racine unique, la classe `Object`. Les hiérarchies sont simples en Java, une classe hérite de la structure et des méthodes de la super-classe. Cependant, une classe peut hériter de multiples interfaces, ce qui permet aux programmes de manipuler les instances selon différents points de vue concernant leur type. Une interface définit des spécifications, sous la forme d'un ensemble de prototypes de méthode, que la classe doit nécessairement implanter [103].

Créer une application en Java consiste donc à écrire de nouvelle classes, par spécialisation de classes existantes (éventuellement de la classe `Object`), à organiser les classes en différents packages liés hiérarchiquement, puis ensuite à définir un ou plusieurs points d'entrée pour l'exécution du programme.

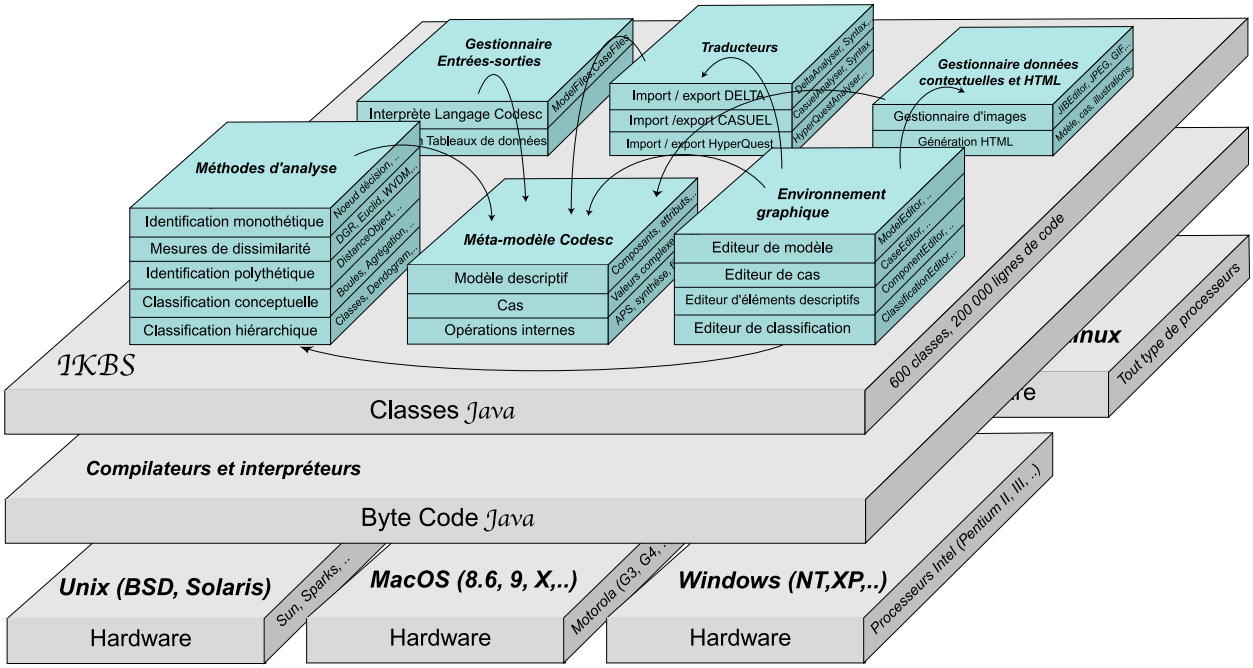


FIG. 6.14 – *Architecture générale d'$\mathcal{IKBS}$.*

103. Cette condition est vérifiée à la compilation

Portabilité

Java est unique en son genre. Il est à la fois un langage *compilé* et *interprété*. Le code source (langage Java) est compilé en *byte-code*, qui est interprété par un *interprèteur Java*. L'avantage principal de ces deux niveaux est de rendre Java indépendant du système d'exploitation hôte (portabilité). Un programme Java peut être exécuté sur tout type de machine (PC, apple, Spark, etc.) et de systèmes d'exploitation (Windows, macOS, Unix, Linux, etc.) comme l'illustre la figure 6.14. $\mathcal{IKBS}$ est donc utilisable sur tout type de système d'exploitation.

Structuration des composants logiciels

$\mathcal{IKBS}$ est donc développé dans le langage Java. Un soin important a été donné à la structuration du code en différents package, dont les plus signifiants sont illustrés à la figure 6.14. Les relations indiquent les dépendances entre les packages.

- Méta-modèle CoDesc. Le système de représentation des connaissances, noyau de la plate-forme (66 classes) ;
- Environnement graphique. Ensemble de classes pour l'édition des connaissances, le paramétrage des méthodes d'analyse, etc. (130 classes) ;
- Méthodes d'analyse. L'ensemble des méthodes d'analyse présentées aux chapitres 4 et 5 (128 classes) ;
- Traducteurs. Regroupe les outils d'importation et d'exportation dans les langages externes (Delta, Casuel, HyperQuest) (54 classes).
- Entrées-sortie. Fonctions liées au chargement des données / connaissances, via le réseau ou en local (26 classes) ;
- Gestionnaire données contextuelles (32 classes) ;
- Classes utilitaires diverses (sécurité, etc.) (environ 50 classes).

La figure 6.15 indique les dépendances entre les packages de manière plus détaillée et introduit le package "serveur" de connaissances. Notons que nous n'avons pas encore à ce jours développé une "architecture client-serveur" pour la gestion collaborative de BC dans le cadre de cette thèse. Cette évolution représente la ligne directrice majeure pour les futurs développements de la plate-forme.

Le package CoDesc

L'ensemble des packages présenté à la figure 6.15 sont dépendants du package de représentation des connaissances. Celui-ci peut être considéré comme le noyau de la plate-forme. Il est indépendant de tout autre package. L'ensemble des classes du package CoDesc forment une hiérarchie unique de racine `Object`, super-classe de toute classe Java.

Trois hiérarchie principales sont considérées dans ce package :

- La classe abstraite `DescriptiveElement`, racine de la hiérarchie d'héritage des éléments constitutifs du modèle descriptif (Composants, Schéma, Attributs, Règles, etc) ;

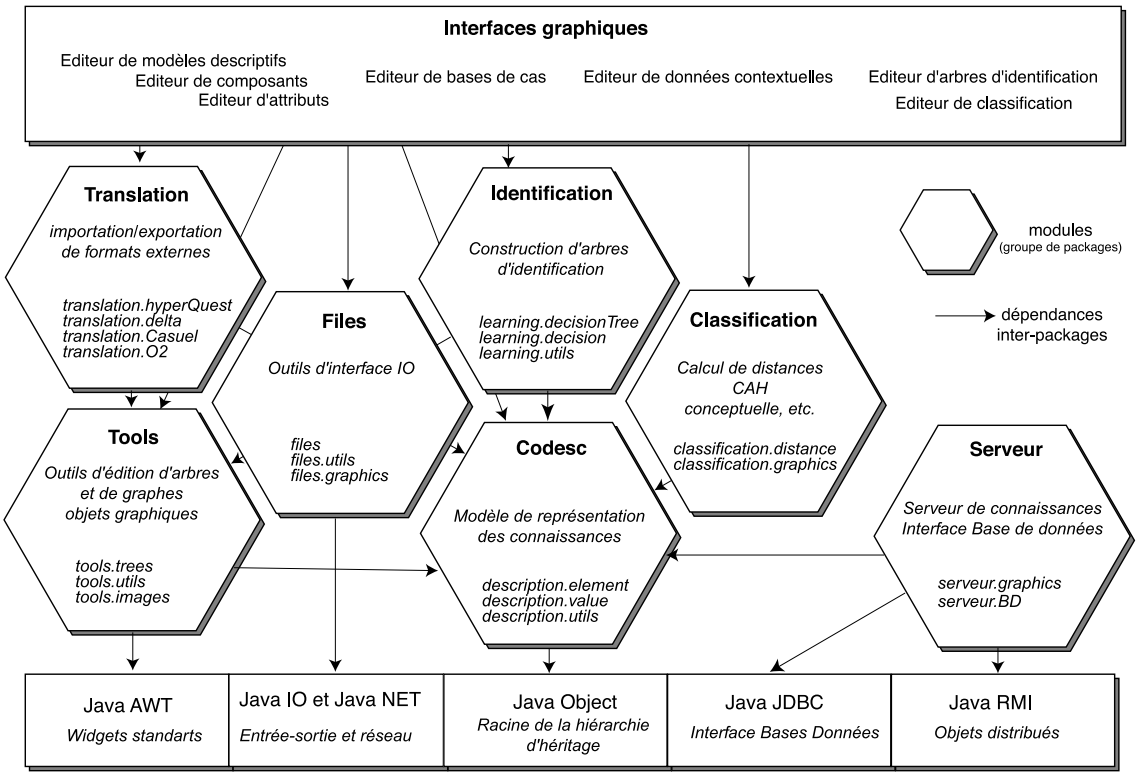


FIG. 6.15 – *Dépendances entre les packages d'$\mathcal{IKBS}$.*

- La classe abstraite `Value`, racine de la hiérarchie de classes pour la représentation des cas. La hiérarchie complète est exposée à l'annexe D ;
- La hiérarchie des évènements (`CodescEvent`) pour la gestion de la dynamique d'évolution des BCs.

6.5.3 Points d'ouverture et axes de variabilité

L'exposé détaillé de la plate-forme $\mathcal{IKBS}$ et des "points d'ouverture" sort du cadre de cette thèse. En particulier, nous avons identifié un certain nombre de "points d'ouverture" (ou "hot spots") (Schmid 1997) [104] selon deux "axes de variabilité" principaux : représentation des connaissances et conception de méthodes d'analyses.

Axe de variabilité "représentation des connaissances"

Pour l'axe de variabilité concernant la représentation des connaissances (le package CoDesc), les points d'ouverture identifiés [105] autorisent en particulier l'extensibilité :

- Des types de descripteurs (au niveau de la hiérarchie `DescriptiveElement`) ;
- Des types de valeurs (au niveau de la hiérarchie `Value`) ;

104. Un point d'ouverture représente un comportement susceptible d'être différent dans les applications visées.
105. Les points d'ouverture identifiés sont à "boîte transparente" : l'ouverture est réalisée par spécialisation des classes existantes.

– Des types d'évènements (au niveau de la hiérarchie `CodescEvent`).

Axe de variabilité "méthodes d'analyse"

Pour l'axe de variabilité concernant la conception de méthodes d'analyse, les points d'ouverture permettent en particulier de définir :

– D'autres mesures de similarité ou de dissimilarité. Nous avons exploité ces points d'ouverture pour concevoir les différentes mesures exposées au chap. 5 ;
– De nouveaux critères de sélection pour la construction de l'arbre d'identification (cf. chap. 4). A l'heure actuelle nous n'avons implanté qu'une seule mesure (gain d'information), d'autres sont possibles (par exemple le "Gini index") ;

6.6 Conclusion

Nous avons exposé dans ce chapitre différents aspects de la plate-forme à objets $\mathcal{IKBS}$, du plus haut niveau (niveau utilisateur) jusqu'au plus bas niveau (niveau informaticien). Partant de la nécessité pour les systématiciens de disposer de méthodes et d'outils informatique, nous avons développé une méthodologie itérative de gestion des connaissances, ainsi que des outils logiciels pour l'acquisition le traitement et la validation des concepts et des descriptions d'un domaine de la Systématique.

Nous avons ensuite présenté globalement l'architecture de la plate-forme $\mathcal{IKBS}$ en introduisant un certain nombre de "points d'ouverture" permettant l'extensibilité du modèle de représentation des connaissances et des méthodes d'analyse associées. $\mathcal{IKBS}$ est ainsi non seulement une application de gestion des connaissances en Systématique, mais également une véritable plate-forme orienté-objets pour la représentation des connaissances descriptives et classificatoires, ainsi que pour la conception et l'expérimentation de méthodes d'analyses sur des données complexes.

L'aspect **itératif** intervient non seulement dans la méthodologie globale de gestion des connaissances (*Knowledge Engineering*), mais également dans le cycle de développement du logiciel (*Software Engineering*). La confrontation de ces deux cycles de vie mettant en oeuvre simultanément un Système de Gestion de Bases de Connaissances pour répondre à des besoins précis d'une part, et une Base de Connaissances visant une finalité bien définie (la réalisation de la "BC Coraux des Mascareignes") d'autre part, constitue l'un des aspects le plus original de notre travail.

Conclusion

Dans le cadre de la recherche de méthodes et d'outils informatiques répondant à la nécessité de préserver le savoir des systématiciens, nous avons développé un modèle conceptuel original de représentation des connaissances, ainsi qu'un ensemble de méthodes d'analyse et d'exploration des connaissances pour l'*identification* et la *classification* des objets biologiques.

Ce modèle et les méthodes associées ont été rendus opérationnels au sein d'une plate-forme logicielle orienté-objets, baptisée $\mathcal{IKBS}$ pour *Iterative Knowledge Base System*. Celle-ci met en oeuvre une méthodologie *itérative* de Gestion des connaissances, fondée sur la méthode scientifique expérimentale des systématiciens, de nature inductive. Elle offre un ensemble d'outils de haut niveau qui facilitent les différentes étapes de construction et de mise au point d'une base de connaissances.

1 Apport de ce travail

1.1 Modèle de représentation

Le modèle de représentation des connaissances CoDesc est une évolution originale des « Logiques Descriptives en Sciences de la Vie » élaborées par N. Conruyt (1994). Il introduit les notions de spécialisation de concepts et de valeurs complexes. En particulier, la dualité concept / instance et la sémantique de la relation de subsomption est inspirée du modèle à objet classique et des logiques terminologiques (propriétés contingentes et nécessaires). Les valeurs complexes offrent le moyen de renseigner des états particuliers dans les descriptions, comme la variation, l'imprécision, l'inconnu ou l'exception, fréquemment observé dans les descriptions d'objets biologiques.

Le formalisme proposé possède également de puissantes opérations internes telles que des opérations formelles sur des *valeurs complexes*, la synthèse de descriptions généralisantes, le filtrage ou la recherche d'Ascendants dans une hiérarchie de concepts. Ces opérations sont a la base des différentes méthodes d'analyse développées pour prendre en compte la complexité des connaissances modélisées dans le formalisme.

Le modèle CoDesc représente à l'heure actuelle l'un des modèles les plus riche et les plus adaptés pour la représentation des concepts spécifiques à la Systématique, et la description de spécimens biologiques.

1.2 Méthodes d'analyse

Notre travail pluridisciplinaire s'est intéressé de près à un certains nombre de domaines des sciences biologiques et informatiques (Analyse de Données, classification, Intelligence Artificielle et Apprentissage Automatique). Nos contributions principales dans ces domaines sont l'élaboration d'une méthode d'identification monothétique interactive et d'une *mesure de dissimilarité* prenant en compte de manière très fine la complexité et la structure des descriptions d'objets biologiques. Cette mesure est à la base d'un ensemble de méthodes d'identification polythétique et de classification (Classification Ascendante Hiérarchique, Nuées Dynamiques Distribuées, Classification Conceptuelle).

La plupart de ces méthodes ont été réalisées en collaboration avec des étudiants [106] ainsi que d'autres collègues chercheurs du laboratoire IREMIA [107].

La plate-forme $\mathcal{IKBS}$ réalisée est un outil de haut niveau développé sous forme de composants logiciels Java, ouverts extensibles et réutilisables. A l'heure actuelle, elle est stabilisée et suffisamment générique pour permettre la construction de nouvelles bases de connaissances dans des domaines proches de la Systématique, où une aide à l'identification et la classification est recherchée. Mais l'application majeure d'$\mathcal{IKBS}$, pour laquelle les travaux de cette thèse ont été initié, concerne principalement la réalisation de la « Base de connaissances sur les coraux des Mascareignes ».

1.3 La réalisation de la « Base de connaissances sur les coraux des Mascareignes »

Depuis plusieurs années, la plate-forme $\mathcal{IKBS}$ a permis à un groupe d'experts français de construire une « base de connaissances sur les coraux des Mascareignes » [108], à partir de près de 3000 spécimens récoltés dans l'Océan Indien et stockés dans la collection G. Faure. Cette base de connaissances est une réponse à l'urgence qu'il y a de préserver le savoir des experts liés à l'observation, la descriptions et la Taxonomie de ces organismes marins.

Cette base de connaissances est davantage qu'un simple recueil de savoir encyclopédique, elle donne également le moyen d'identifier avec une précision de 80% l'espèce à laquelle appartient un échantillon relevé dans la nature [109].

Dans cette thèse, nous n'avons pas présenté intégralement cette réalisation, mais les principaux problèmes liés à la représentation de ces objets biologiques atypiques, vivants en colonies et par nature fortement polymorphes, ont été clairement exposés. Un ensemble d'expérimentations a été effectué dans le but de valider la robustesse globale de la base de connaissances coraux.

Le fruit de ce travail est disponible sur Internet [110] afin de la rendre accessible à un large public, amateurs ou professionnels.

106. Sur les cinq dernières années, nous avons encadré 17 étudiants de la maîtrise d'informatique de l'Université de la Réunion.

107. Je remercie chaleureusement N. Conruyt, J. Diatta et H. Ralambondrainy pour leur constant intérêt et leur contribution à ces travaux.

108. Le projet a été financé par la Région Réunion.

109. Les douaniers en particulier ont parfois besoin d'identifier des spécimens, certaines espèces étant protégées par la "convention de Washington" et ne peuvent donc être déplaçées.

110. *http ://www.univ-reunion.fr/coraux*

Pour cette réalisation, notre contribution a été principalement liée à la conception et au développement d'$\mathcal{IKBS}$, l'outil informatique clef pour supporter le travail des spécialistes, qui représente un effort constant de près de cinq années de conception et de programmation. Nous avons également offert une assistance aux experts quant à son utilisation et apporté à mesure de nombreuses améliorations, tant en ce qui concerne les fonctionalités que la convivialité de son utilisation.

Nous considérer que la « Bases de Connaissances coraux » est au coeur même de processus d'élaboration de la plate-forme réalisée. La confrontation synergétique permanente entre des besoins réels exprimés par les systématiciens (côté usages) et les solutions technologiques apportées par les informaticiens (côté métier) a été une source intarissable pour de nouvelles idées et de nouvelles fonctionnalités.

2 Perspectives

Plusieurs projets de BC sont actuellement en cours d'élaboration à l'aide de la plate-forme $\mathcal{IKBS}$:

- Aide au diagnostique médical en médecine nucléaire : une application en Coronaropathie (maladies coronariennes) fondée sur l'interprétation des caractères distinctifs à partir d'analyse d'images scintigraphiques cardiaque est en cours ;
- Identification et classification des pathologies mentales : projet en émergence avec l'association française de neuropathologie ;
- Projet Mauriflor@ : certaines familles de plantes endémique de l'île Maurice sont encore pas / peu étudiées. Il semble qu'elles possèdent de très bonnes propriétés antioxydantes et présentent donc des applications pharmaceutiques très intéressantes. Une convention a été signée avec l'Université de Maurice pour développer ce projet avec $\mathcal{IKBS}$.
- Famille des Sertulariidae : cette famille d'organismes marins très particulier (les Hydraires) a été modélisé avec $\mathcal{IKBS}$. Par ailleurs, un projet de collaboration est en cours en collaboration avec le Laboratoire Informatique et Systématique de Paris 6 [111] pour l'informatisation à l'aide des Bases de Données Orientées Objets de la Systématique de cette famille. Ce travail devrait permettre l'interopérabilité des SGBDOOs avec les bases de connaissances.

111. Thèse de G. Rousse en cours.

Cinquième partie

Les annexes

Annexe A

Clés interactives et $\mathcal{IKBS}$

A.1 Introduction

Dallwitz et al. (2000) souligne l'avantage fondamental des *clefs interactives* offert par les systèmes d'Identification Assistée par Ordinateur (I.A.O.) par rapport aux *clefs conventionnelles* (clefs papier), en mettant en avant les caractéristiques suivantes :

A.1.1 Caractéristiques fondamentales des systèmes d'I.A.O.

- **Robustesse**. Une identification correcte peut-être effectuée, même si l'utilisateur fait des erreurs ou que certaines données sont erronées.
- **Souplesse**. N'importe quel caractère peut être utilisé et sa valeur changée, dans n'importe quel ordre.
- **Caractères numériques**. Les caractères numériques peuvent être renseignés directement, sans être discrétisés en intervalles.
- **Incertitude**. L'utilisateur peut exprimer l'incertitude en entrant plusieurs valeurs ou un intervalle de valeurs.
- **Mise à jour**. La mise à jour est simplifiée en modifiant les données, alors que la mise au point de clefs conventionnelles est relativement difficile. En effet, l'ajout de nouveaux caractères ou de nouveaux taxons peuvent amener à reconsidérer entièrement la clef manuelle.

D'autre part, certaines caractéristiques, moins importantes, peuvent être prises en compte lors de la réalisation d'un systèmes d'I.A.O. :

A.1.2 Caractéristiques secondaires

- Avertir l'utilisateur du caractère le plus pertinent à chaque étape de l'identification.
- Localiser l'origine des erreurs par le mécanisme de tolérance aux erreurs.
- Dépendance des caractères : certaines valeurs peuvent rendre d'autres attributs inapplicables.
- Stocker, rechercher et afficher des informations libres.

- Utilisation des probabilités.
- Présence d'un glossaire et de notes sur l'interprétation des caractères.
- Trouver les différences et les similarités entre taxons.
- Trouver des descriptions diagnostiques.
- Illustrations des caractères et des taxons.
- Capacité à gérer de manière efficace d'importantes quantités de données.
- Partager des données avec d'autres systèmes basés sur les descriptions.

A.2 Tableau de comparaison

Le tableau suivant présente de manière détaillée les caractéristiques attendues d'un système d'I.A.O., d'après l'auteur de INTKEY (Dallwitz et al. 1995), l'un des systèmes d'I.A.O. les plus utilisés à ce jours par les biologistes, qui pour sa part répond à l'ensemble de ces caractéristiques désirables.

Nous présentons ces caractéristiques en 67 points, tels qu'ils sont exposés dans Dallwitz et al. (2000), en mettant en évidence leur présence ou leur absence au sein d'$\mathcal{IKBS}$. Nous montrons ainsi que notre système d'I.A.O. répond pour sa part à la plupart d'entre elles.

n°	**Caractéristiques attendues**	**Commentaires**	*$\mathcal{IKBS}$*
Stratégies			
1	Conserver l'inconnu	Les taxons pour lesquels un caractère n'est pas connu sont retenus lorsque le caractère est utilisé. Ceci est essentiel pour une identification correcte.	OUI
2	Utilisation non-restrictive des caractères	Aucune restriction concernant l'ordre d'utilisation des caractères, mise à part les restrictions imposées par les caractères dépendants (voir ci-dessous).	OUI
3	Changer ou effacer les caractères	Pouvoir à tout moment changer ou effacer les valeurs renseignées. Caractéristique non-désirée : ne pouvoir changer que dans l'ordre inverse.	OUI
4	Tolérance aux erreurs	Possibilité d'accéder à une identification correcte, même si les données sont erronnées ou que l'utilisateur se trompe.	Cela dépend fortement des données.
5	Localiser les erreurs	Lorsque la tolérance aux erreurs n'est pas nulle, le programme doit distinguer celles provenant des données de celles provenant de l'utilisateur.	NON
6	Exprimer l'incertitude	L'utilisateur peut exprimer l'incertitude en renseignant un ensemble de valeurs ou un intervalle.	OUI
7	Caractères numériques	Utiliser directement les caractères numériques sans les diviser en intervalles.	OUI
8	Ensemble ou intervalles	Les caractères numériques doivent pouvoir être renseignés sous forme d'ensembles : 5 ou 10 plutôt que [5 10].	OUI

Choix des caractères

9	Meilleur caractère	Renseigner le meilleur caractère à toute étape de l'identification.	OUI
10	Séparation des taxons	Classer les caractères en fonction de leur pouvoir séparateur d'un taxon désigné par rapport aux autres.	NON
11	Différencier les attributs	Différencier les attributs caractéristiques des taxons de ceux qui ne le sont pas.	OUI, par le gain d'information
12	Pondération des caractères	Des poids peuvent être associés aux attributs pour le calcul du meilleur attribut. Limitation indésirable : les plus hauts poids sont toujours les meilleurs.	OUI. Les poids associés aux caractères sont combinés avec le pouvoir discriminant

Différents types de valeurs

13	Valeurs inconnues	Possibilité de renseigner la valeur inconnue dans les données.	OUI
14	Dépendance des caractères	Relations spécifiants que certains caractères sont inapplicables lorsque d'autres caractères prennent certaines valeurs	OUI. De deux manières, par les relations de compositions ou par les règles.
15	Contrôle automatique des caractères	Dans quelques cas, il est souhaitable de pouvoir renseigner la valeur d'un caractère qui a été rendu inapplicable par un autre caractère dont il dépend.	Pas vraiment. L'utilisateur peut désactiver l'option de gestion automatique des dépendances, mais pas pour un caractère particulier.
16	Caractères de type Texte	Stocker de l'information de type textuelle concernant les données et les caractères.	OUI
17	Valeurs spéciales pour les clefs	L'utilisateur des clefs doit pouvoir affecter une valeur non prévues, inexistante dans les données.	OUI, par utilisation des valeurs exceptionnelles.
18	Identification probabiliste	Combiner les probabilités de présence des valeurs pour un taxon donné avec celles des erreurs de l'utilisateur, pour calculer la probabilité qu'un spécimen appartiennent à un taxon donné.	NON
19	Inapplicable contre inconnu	Distinguer l'inapplicable de l'inconnu.	OUI
20	Extension aux intervalles pour les caractères numériques	Traiter les valeurs numériques simples comme des intervalles. Limitation indésirable : la transformation n'est pas sous le contrôle de l'utilisateur de la clef	NON, même si les valeurs numériques peuvent être transformées en intervalles.

Sous-ensembles

21	Nommer des sous-ensembles de caractères	Un mécanisme pour nommer les sous-ensembles de caractères. Limitation indésirable : les sous-ensembles ne sont pas nommés par l'utilisateur pendant la phase d'identification.	NON, mais des sous-groupes de caractères sont formés par les objets auxquels ils sont attachés.
22	Sous-ensembles de caractères pour le traitement	Spécifier des sous-ensembles de caractères qui ne seront pas traités par l'identification	OUI, par l'intermédiaire des poids.
23	Sous-ensembles de caractères locaux	Spécifier des sous-ensembles de caractères qui ne seront pas traités par une opération simple de l'identification	NON

24	Nommer des sous-ensembles de taxons	Mécanisme pour nommer des sous-ensembles de taxons	NON, la structuration en hiérarchie de valeurs permet de prédéfinir des sous-groupes par niveaux taxonomiques
25	Définir des sous-ensembles de taxons	pour lesquelles l'identification sera restreinte	OUI, en conservant uniquement les taxons qui nous intéressent.
26	Sous ensembles locaux de taxons	Définir des sous-ensembles de taxons pour lesquels une opération simple ne sera pas effectuée.	NON

Interprétation des caractères

27	Annotations sur les caractères	Aider à la compréhension et l'interprétation des caractères.	OUI
28	Glossaire	Liens vers la définition des mots dans la base	OUI, pas au sein du système, mais par l'intermédiaire des liens hypertextuels.
29	Illustration des caractères	Afficher les illustrations des caractères.	OUI
30	Sélection de valeurs par l'intermédiaire des illustrations	Choisir une illustration renseigne la valeur	OUI

Images et sons

31	Illustrations des taxons	Afficher les illustrations des taxons pendant l'identification	OUI
32	Illustrations des taxons par sujet	Sélectionner les illustrations des taxons en fonction du sujet (ex. : fleurs, habitat, distribution, etc.)	NON
33	Utilisation flexible des illustrations	Les illustrations peuvent être de toute taille, redimensionnées, affichées simultanément, etc.	OUI
34	Texte avec illustrations	Du texte peut être associé aux illustrations.	OUI
35	Sons	Du son peut être attaché aux caractères et aux taxons	OUI
36	Vidéos	Des vidéos peuvent être associées aux caractères et aux taxons	OUI
37	Fonctionnement sans les illustrations	L'application peut fonctionner sans les illustrations associées	OUI

Recherche d'information

38	Recherche dans la liste des caractères	Rechercher par chaînes de caractères (String) dans la liste des caractères (attributs).	NON, mais la liste est accessible par ordre alphabétique.
39	Recherche de taxons	Rechercher des chaînes de caractères dans la liste des taxons et des synonymes.	NON
40	Fixer la valeur de certains caractères	Fixer la valeur de certains caractères connus à l'avance	OUI
41	Sous-clefs intégrées	Avoir plusieurs clefs disponibles pour le même jeux de données	OUI
42	Sous-clefs séparées	Lier les sous-clefs de niveaux taxonomiques différents. Par exemple, liens entre les clefs de genre et les clefs des espèces appartenant au même genre	OUI

43	Afficher les descriptions	Afficher les descriptions utilisées pour l'identification	OUI
44	Différences entre taxons	Mettre en avant les différences entre les membres d'un ensemble de taxons	NON. Possiblité d'afficher ce qui est similaire, mais pas ce qui est différent.
45	Similarité entre taxons	Ressemblance entre membres d'un ensemble de taxons.	OUI
46	Descriptions diagnostiques	Trouver des descriptions diagnostiques. Une description diagnostique pour un taxon correspond au taxon, mais diffère de tout autre taxon sur la donnée d'au moins un caractère.	OUI, par synthèse des descriptions appartenant au taxon
47	Recherche multi-critères de taxons	Trouver les taxons qui correspondent à une combinaison de critères.	OUI
48	Distribution des valeurs	Afficher la distribution des valeurs en fonction des taxons	NON
49	Taxons les plus similaires	Classer les taxons par similarité par rapport à un taxon donné	OUI

Partage des données

50	Importer le format DELTA	Uiliser le format DELTA pour générer des clefs interactives	OUI, mais encore à mettre au point
51	Exporter le format DELTA	Exporter le format DELTA	OUI, encore à mettre au point
52	Production de descriptions papier	Générer des descriptions dans un format publiable à partir des données utilisées par le système d'identification.	OUI, au format HTML
53	Sortie de données	Sortie les données dans un format utilisable par d'autres logiciels d'analyse.	OUI, tableaux de données
54	Génération de clefs conventionnelles	Générer des clefs conventionnelles à partir des données utilisées par le système d'identification.	OUI, arbre de décision statique
55	Lien avec la classification	Effectuer des classifications cladistiques et phénétiques à partir des données.	OUI, mais uniquement phénétiques

Facilité d'utilisation

56	Aide hypertexte	Aide en ligne complète et sensible au contexte.	OUI, mais pas totalement réalisé
57	Lignes de commandes ou macro	Mécanisme pour répéter une série d'opérations.	NON
58	Barre d'outils adaptable	L'utilisateur peut définir sa propre barre d'outils pour les opérations les plus fréquentes.	NON
59	Ressources textuelles externes	Les ressources du système dépendantes de la langue sont externalisées afin de permettre la création de différentes versions.	OUI
60	Fichier de log	Création d'un historique automatique.	NON, pas encore
61	Pas de limites de taille des données	Le nombre de taxons et de caractères n'est pas limité.	OUI
62	Pas de limites des champs de saisie	Longueur des champs non limité (nom des taxons, annotations, etc.).	OUI
63	Occupation mémoire raisonnable	Le programme doit pouvoir fonctionner avec un minimum de mémoire, normalement requis par le système d'exploitation.	OUI

64	Vitesse d'exécution	Le temps d'exécution des opérations les plus représentatives doit être raisonnable sur un ensemble de données assez large : 400 taxons, 200 caractères.	OUI
65	Fonctionnalité Internet	Le programme peut accéder à des données et des images via Internet.	OUI
66	Installation non nécessaire	Le programme peut être directement utilisé depuis un CD-ROM, sans installation nécessaire.	OUI, si un interprète Java est installé
67	Interface utilisateur simple	L'interface utilisateur est simple, efficace et conviviale	OUI

TAB. A.1 – *Tableau de comparaison des fonctionalités attendues d'un système d'I.A.O. avec IKBS*

Annexe B

Illustrations de la famille des Pocilloporidae

Cette section présente quelques illustrations que nous avons élaborées pour la la base de connaissances coraux. Elles servent de guide pour la description des échantillons, pour renseigner ou pour identifier un nouveau cas à l'aide des clefs d'identification générées par $\mathcal{IKBS}$.

La totalité des illustrations sont accessibles à l'URL : *http ://www.univ-reunion.fr/ ikbs*

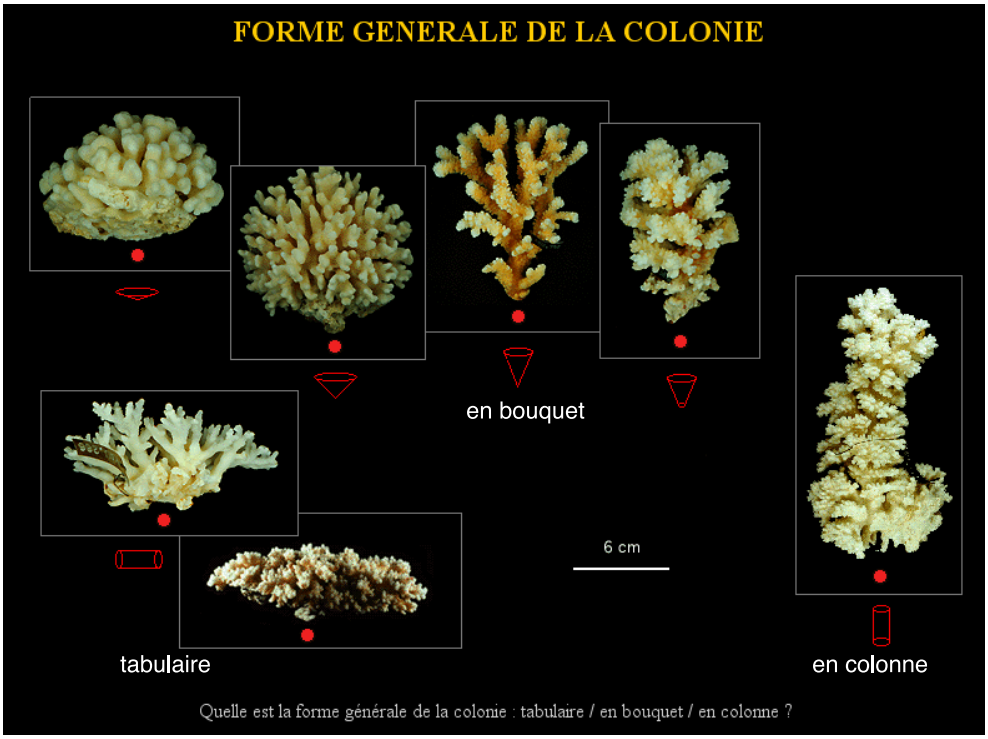


FIG. B.1 – ***Question*** *: Quelle est la forme générale de la colonie : tabulaire, en bouquet ou en colonne ?* ***Commentaire*** *: Il faut repérer le point de fixation de la colonie, puis, par rapport à ce point de fixation, déterminer la forme générale de la colonie. Forme en bouquet : court ($H/L < 1$), sphérique ($H/L = 1$), dressé ($H/L > 1$). Attention : on considère la forme de la colonie toute entière et non celle d'un fragment de colonie.*

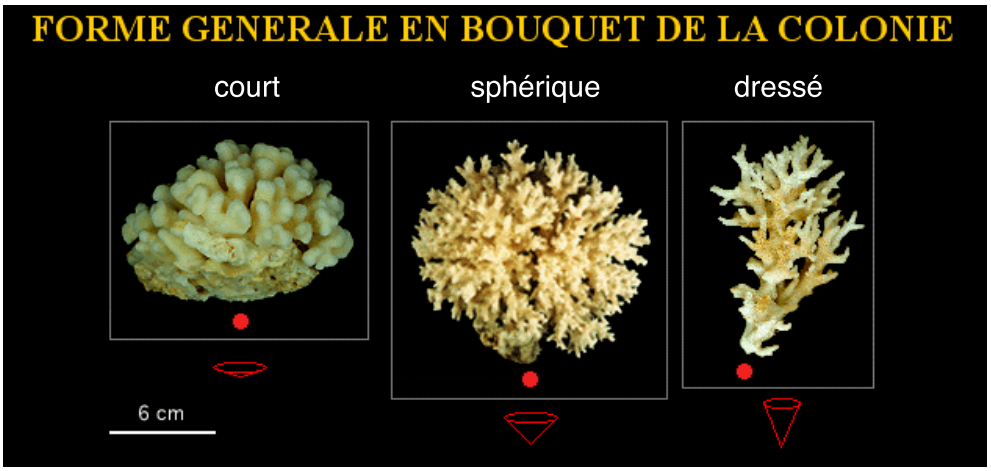


FIG. B.2 – ***Question*** *: Si la forme générale de la colonie est en bouquet, est-elle de type : court, sphérique ou dressé ?* ***Commentaire*** *: court ($H/L < 1$), sphérique ($H/L = 1$), dressé ($H/L > 1$).*

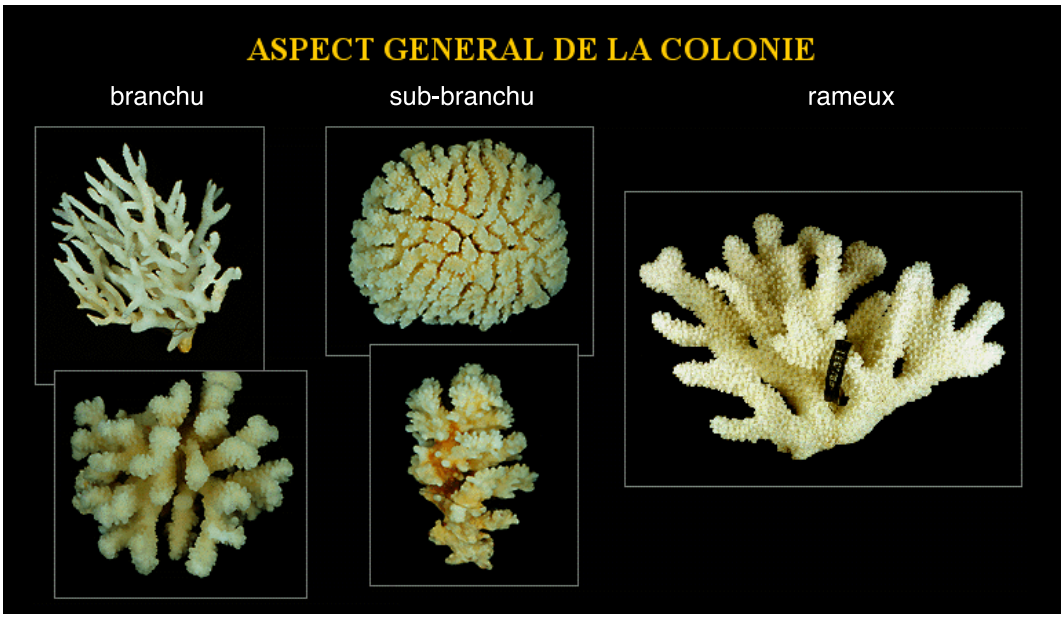


FIG. B.3 – ***Question*** *: Quel est l'aspect générale de la colonie : branchu, sub-branchu ou rameux ?* ***Commentaire*** *: rameux est un cas particulier de l'aspect branchu dans lequel les branches sont espacées et dont l'extrémité est à la fois aplatie et élargie en forme de rame.*

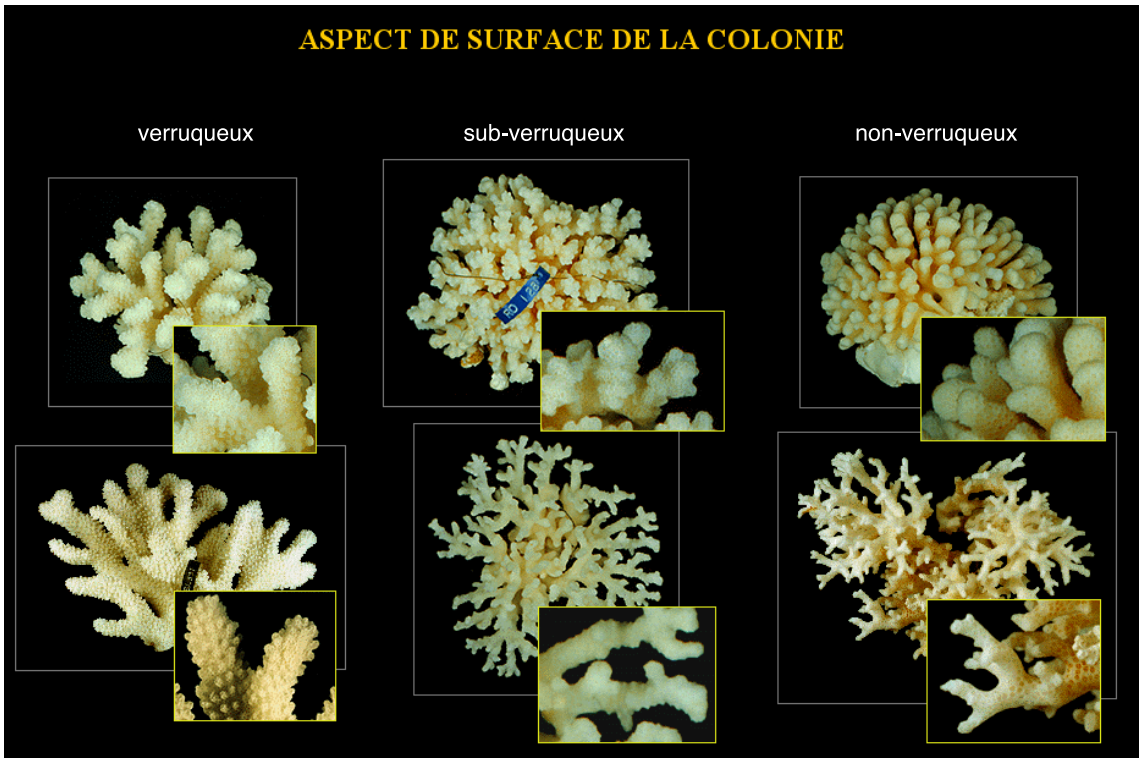


FIG. B.4 – ***Question*** *: Quel est l'aspect de surface de la colonie : verruqueux, sub-verruqueux ou non-verruqueux ?* ***Commentaire*** *: pour la valeur « sub-verruqueux », il est impossible de distinguer d'une part les vraies verrues, d'autre part la naissance des branches secondaires.*

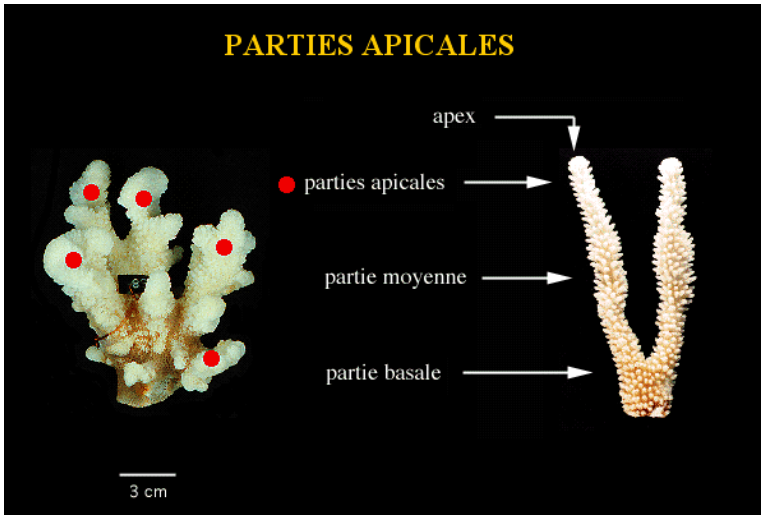


FIG. B.5 – *Les parties apicales des branches correspondent à la région distale ou supérieure de ces dernières (et non pas de la colonie toute entière). L'apex définit l'extrême partie sommitale de ces mêmes branches.* ***Attention*** *: en cas de description d'un fragment de colonie, les données quantitatives (diamètre) se rapportent évidemment à celui-ci et non pas à la totalité de la colonie. Il est cependant recommandé de faire figurer les mesures qui sont référables à la colonie toute entière, lorsque cela est possible (par exemple, mesures prises in situ). Dans ce dernier cas, les données quantitatives seront celles de la colonie entière. Toutefois, lorsque la taille du fragment ne permet pas de situer celui-ci à l'échelle de la colonie entière, on préfèrera ne pas donner de réponse.*

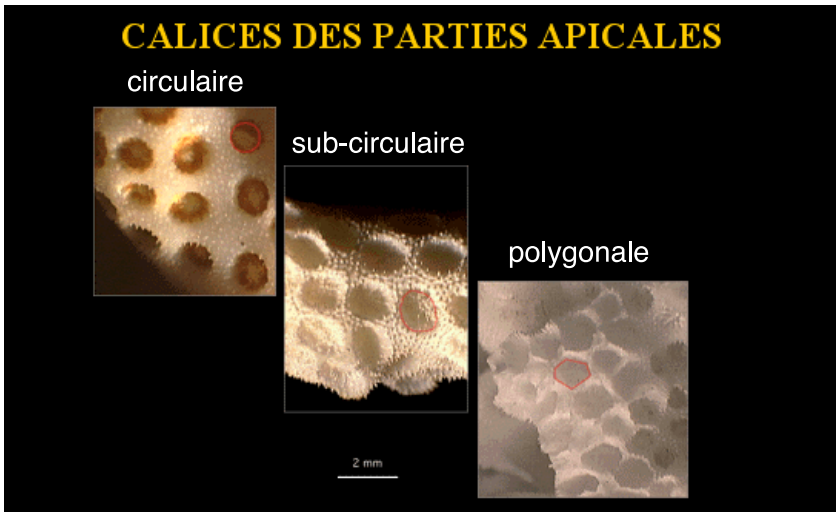


FIG. B.6 – ***Question*** *: Quelle est la forme des calices des parties apicales des branches : circulaire, sub-circulaire ou polygonale ?*

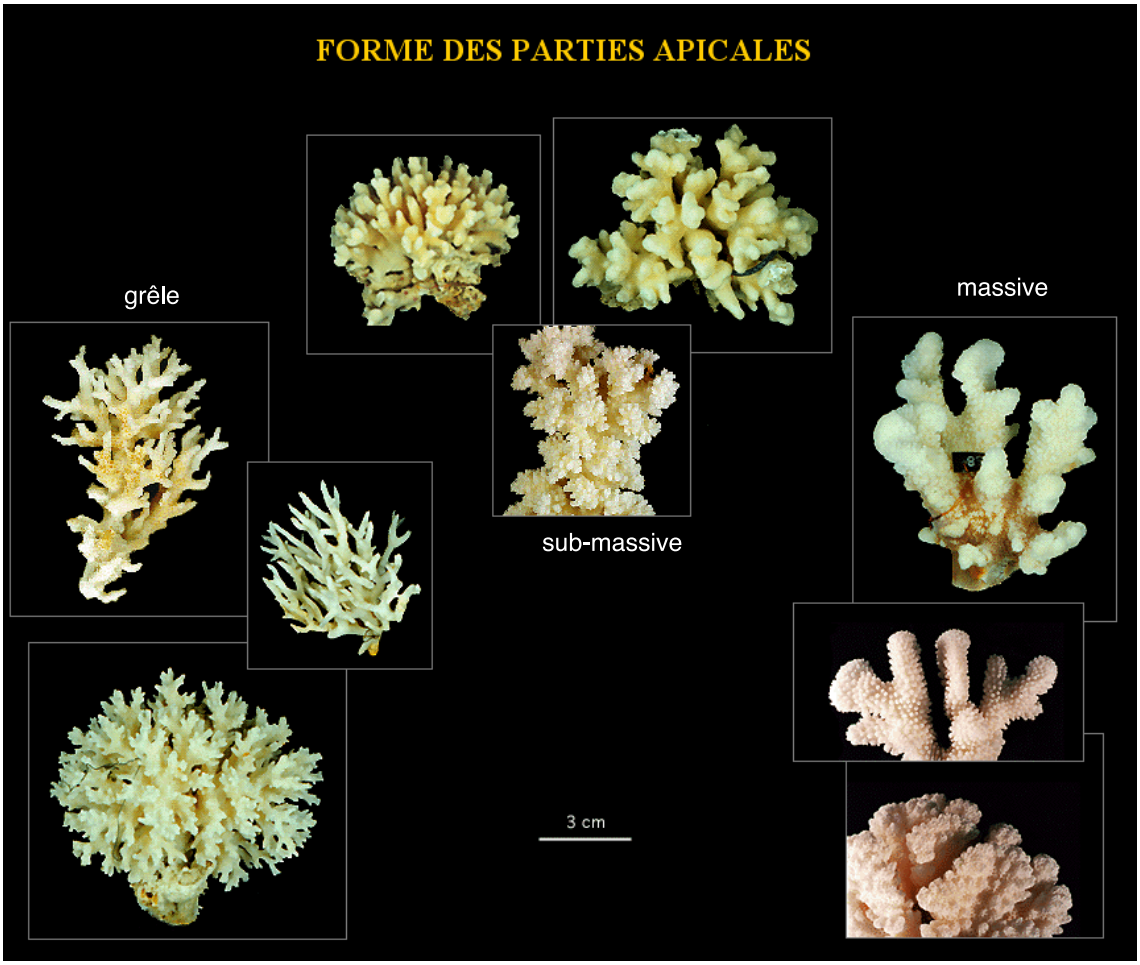


FIG. B.7 – ***Question*** *: Quelle est la forme des parties apicales des branches de la colonie : grêle, sub-massive ou massive ?*

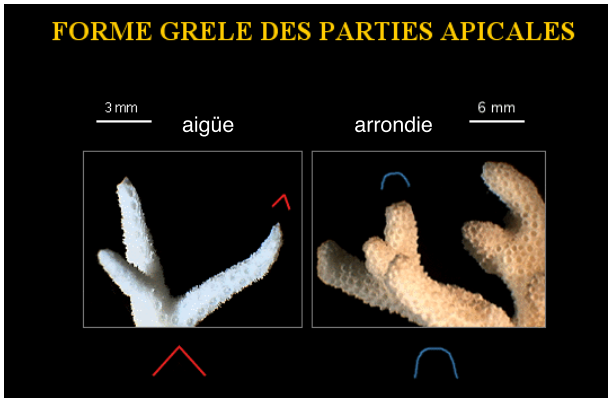


FIG. B.8 – ***Question*** *: Si la forme des parties apicales des branches de la colonie est grêle, est-elle de type : aigue ou arrondie ?*

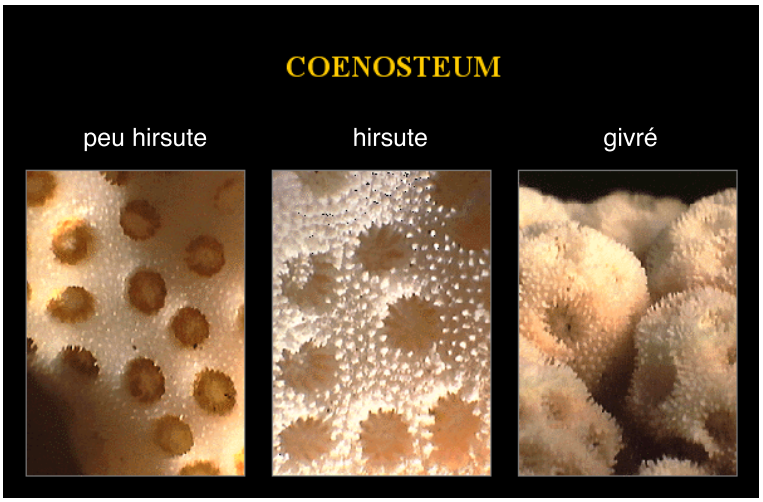


FIG. B.9 – ***Question*** *: Quel est l'aspect général du coenosteum : peu hirsute, hirsute ou givré ?* ***Commentaire*** *: l'aspect général est fonction de la localisation, du grossissement (échelle d'observation), et de l'éclairage (intensité, orientation). Il dépend aussi du plus ou moins grand développement des épines coenostéales (taille) et de la forme de leur extrémité.*

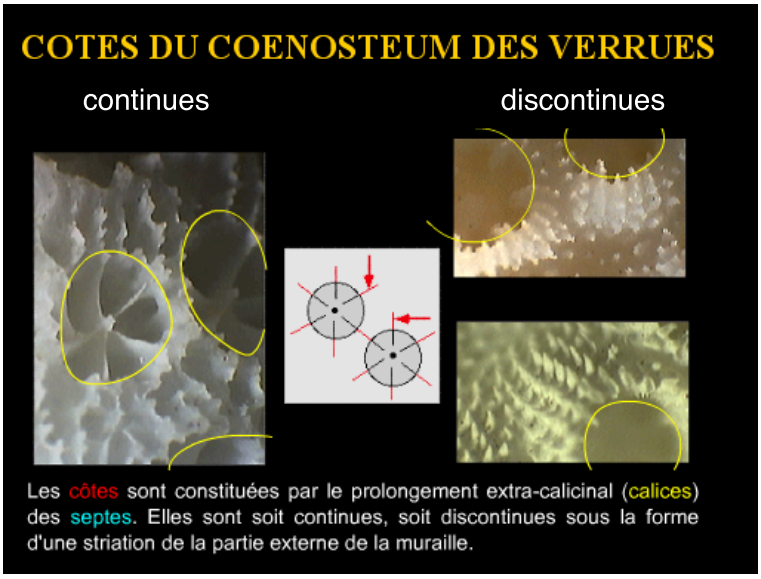


FIG. B.10 – ***Question*** *: Les côtes du coenosteum sont elles continues ou discontinues ?*

Annexe C

Syntaxe B.N.F. du Langage Externe de Représentation des Connaissances

Nous décrivons dans cette annexe la syntaxe du Langage externe de Représentation des Connaissances (LRCO) de CoDesc. Nous l'avons joint en annexe, pour les utilisateurs d'$\mathcal{IKBS}$, car il est parfois utile de pouvoir éditer/modifier les connaissances directement dans ce langage, notament, lors de modifications d'un ensemble de nom, à l'aide d'un éditeur de texte standart.

Nous donnons la définition en grammaire B.N.F. [112] du LRC, qui est utilisée essentiellement dans deux situations, pour :

- exprimer/modifier les connaissances sous forme textuelle,
- gérer la persistance des données/connaissances par le système $\mathcal{IKBS}$.

Les expressions sont stockées dans deux types de fichier, l'un pour la représentation du modèle descriptif, l'autre pour la base de cas. Ces fichiers sont encodés au format universel ISO-8859-1 afin de garantir la compatibilité sur tout type de plate-forme.

Le langage se présente sous la forme de différentes rubriques commençant par des déclarations de définition, du type `Def_Object`, `Def_Attribute`, etc.

C.1 Notes sur la syntaxe BNF du LRCO

Les parties en gras correspondent aux mots-clefs et délimiteurs du langage. [, , *, + sont les marques syntaxiques de la grammaire BNF. <>indique une rubrique détaillée par la suite.

En résumé :

- x* : 0 ou plusieurs occurrences de x,
- x+ : 1 ou plusieurs occurrences de x,

x : 0 ou 1 occurrence de x,

- x | y : soit x soit y

112. Backus-Naur-Form

C.2 Syntaxe du modèle descriptif

```
<Element>       ::= <Schema> | <Component> | <Attribute>

<Schema>        ::= (Def_Schema <identificateur du Schema>
                        Name = <label du schema>
                        [Super = <identificateur du super-objet>]
                        [Subparts_components = identificateur de composant*]
                        [Subparts_attributes = identificateur d'attribut*]
                        [URL = adresse_url*]
                        [Question = String]
                        [Commentary = String]
                        [Illustrations = <Illustrations>* | <Index>])
            )

<Component>     ::= (Def_component <identificateur de l'objet>
                        Name = <label de l'objet>
                        [Super = <identificateur du super-objet>]
                        [Subparts_components = identificateur de composant*]
                        [Subparts_attributes = identificateur d'attribut*]
                        [Fictituous = true | false]
                        [Multiple = true | false]
                        [PossiblyAbsent = true | false]
                        [URL = <Path>*]
                        [Question = String]
                        [Commentary = String]
                        [Illustrations = <Illustrations>* | <Index>
                    )

<Attribute>     ::= (Def_Attribute <identificateur de l'attribut>
                        Name = <label de l'attribut>
                        Type = Symbolic | Taxonomic | Numeric | Textual
                        [Domain = <range>]
                        [URL = adresse_url *]
                        [Question = String]
                        [Commentary = String]
                        [Illustrations = <Illustrations>* | <Index>]
                    )

<Range>         ::= <nominal range> | <taxonomical range> | <numerical range>

<nominal range>     ::= <symbol>+
```

```
<symbol>            ::= String

<taxonomical range> ::= <taxo_symbol>+

<taxo_symbol>       ::= <super_symbol> | <symbol>+

<super_symbol>      ::= String

<numerical range>   ::= [ <value> <value>]

<value>             ::= Integer | Float


<Illustrations>     ::= [<Path> <Relative> <Value>]+
<Path>              ::= String ;; Chemin d'accès ou URL
<Relative>          ::= true | false ;; Chemin d'accès absolu ou relatif
<Index>             ::= (Def_Index
                           Path = <chemin d'accès relatif ou URL>
                           Relative = true | false
                         )
```

C.3 Syntaxe des cas

```
<cas>               ::= <case_number> [<case_value>]*

<case_number>       ::= Integer

<case_value>        ::= <Element> <Value> [<Illustrations>]*

<Value>             ::=       e |                    ;; Valeur non renseignée
              ? |                       ;; Valeur inconnue
                                <symbol > |            ;; Symbole simple
                                <taxo_symbol> |        ;; Symbole taxonomique
             [<symbol> & <case_value>] ;; Valeur conjonctive
             [<symbol> | <case_value>] ;; Valeur disjonctive (ensemble)
```

Annexe D

Diagramme UML pour la représentation des cas

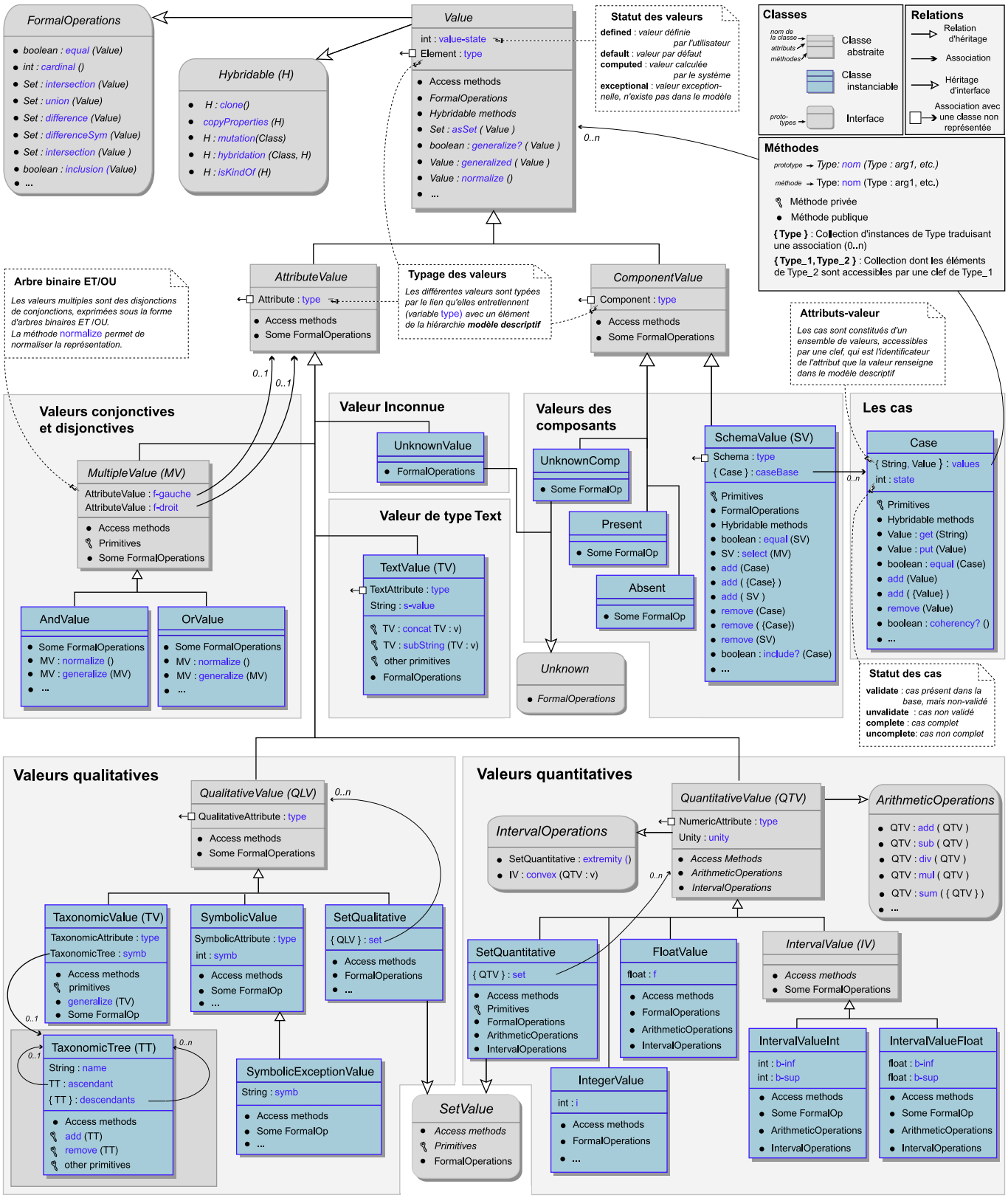


FIG. D.1 – *Diagramme de classes pour la représentation des cas*

Annexe E

Quelques liens utiles à la compréhension du domaine

E.1 Gestion et représentation des connaissances

- **GRACQ** - Groupe de Travail en Acquisition et Ingénieurie des Connaissances.
  http ://www.irit.fr/GRACQ
- **IDEM** - Projet « Images et Diagnostic par l'Exemple en Médecine » de l'hôpital Broussais de Paris.
  http ://www.hbroussais.fr/Broussais/InforMed/IDEM/1Page1.html
- **SHERPA** - Projet « Bases de connaissances à objets » de l'INRIA Rhône-Alpes. Projet stoppé.
  http ://www.inrialpes.fr/sherpa/sherpa-fra.html
- **ROMANS** - « Relations et Objets pour la Modélisation Alliant Numérique et Symbolique ». Projet de l'INRIA Rhône-Alpes.
  http ://www.inrialpes.fr/romans/

E.2 Web sémantique, interopérabilité

- **EXMO** - « Computer mediated Exchange of structured knowledge ». Projet de l'INRIA Rhône-Alpes.
  http ://www.inrialpes.fr/exmo/
- **Semantic Web Community**
  http ://www.semanticweb.org/

E.3 Analyse de données et classification

- **IFCS** - International Federation of Classification Societies
  http ://edfu.lis.uiuc.edu/ class/ifcs/
- **SFC** - Société Francophone de Classification *http ://www.fundp.ac.be/ mremon/sfc/index.shtml*

E.4 Informatique et systématique

- **RNB** - Réseau National de Biosystématique

http ://rnb.snv.jussieu.fr/

- **RNB** - la systématique sur le web - Liens vers des serveurs HTTP en systématique
  http ://rnb.snv.jussieu.fr/Liens/LiensSyst.html
- **ETI** - Expert center for Taxonomic Identification
  http ://wwweti.eti.bio.uva.nl/
- **BIOSIS** - Ressources sur les sciences de la vie
  http ://www.biosis.org.uk/
- **BBCWS** - Biodiversity and Biological Collections Web Server
  http ://biodiversity.uno.edu/
- Specify - Système de gestion de collection évolué
  http ://usobi.org/specify/

E.5 Les grands projets internationaux liés à la Systématique

- **SAS2000** - Systematic Agenda 2000
  http ://www.nhm.ac.uk/esf/leidweb.html
- **SBN** - Systematic Biology Network
  http ://www.esf.org/ln/ln_ 005.htm
- **GBIF** - Global Biodiversity Information Facility
  http ://www.oecd.org/dsti/sti/s_ t/ms/prod/gbif.htm
- **DABS** - Database Activities in the Biological Sciences
  http ://www.nsf.gov/bio/pubs/bir_ dba/dba9606.htm

E.6 Les coraux sur le web

- **ICRI** - Initiative internationale pour les récifs coralliens
  http ://www.environnement.gouv.fr/icri/
- **IFRECOR** - Initiative Française sur les récifs coralliens
  http ://www.environnement.gouv.fr/ifrecor/default.htm
- **GCRMN** - Global Coral Reef Monitoring Network
  http ://coral.aoml.noaa.gov/gcrmn/index.html
- **Reef Check** - Programme international de surveillance des récifs coralliens
  http ://www.reefcheck.org/
- **Reef Base** - base de données sur les coraux de l'ICLARM
  http ://www.reefbase.org/
- Hawai'i Coral Reef Network - Infos sur les récifs coralliens
  http ://www.coralreefs.hawaii.edu/ReefNetwork/default.htm
- AIMS - Institut Australien des Sciences Marines
  http ://www.aims.gov.au

Annexe F

Liste complète de nos publications

Année 2001

CMN'01 "La télésystématique : contribution à la création de bases de connaissances multi-expertes en médecine nucléaire avec IKBS" (résumé). N. Conruyt, H. De Clermont, D. Grosser, N. Robert, F. Salmon. 39ème Colloque de Médecine Nucléaire de langue française, vol. 25, n°10, p. 574, la Réunion (France), 2001

CMN'01 "Etude préliminaire à l'utilisation de la plate-forme IKBS pour la création de bases de connaissances en Médecine Nucléaire : approche en cardiologie" (résumé). H. De Clermont, N. Conruyt, D. Grosser, N. Robert. 39ème Colloque de Médecine Nucléaire de langue française, vol. 25, n°10, p. 591, la Réunion (France), 2001

Année 2000

ICRS'00 "A dissimilarity measure for structured descriptions in natural sciences : Application to sponge and coral systematics". N. Conruyt, D. Grosser, J. Le Renard. 9^{th} Int. Conf. for Reef Studies, Bali, 2000

ICRS'00 "IKBS, A tool for building cooperative knowledge bases on the internet : application to corals and hydroids of the Mascarene archipelago" (poster). N. Conruyt, D. Grosser, Y. Geynet, G. Faure, M. Pichon, N. Gravier-Bonnet, M. Guillaume. 9^{th} Int. Conf. for Reefs Studies, Bali, 2000

ICRS'00 "Object-oriented Systematics Information System. Toward systematics domain objects" (poster). G. Rousse, D. Grosser, N. Conruyt, R. Vignes, M. Pichon, N. Gravier-Bonnet, M. Guillaume. 9^{th} Int. Conf. for Reefs Studies, Bali, 2000

PAKDD'2000 "Improving Dissimilarity Functions with Domain Knowledge, applications with IKBS system". D. Grosser, J. Diatta, N. Conruyt. 4th. Int. conf. on Practical Applications of Knowledge Discovery in Database, D.A. Zighed, J. Komorowski, J. Zytkow (Eds.), LNAI 1910, pp. 409-415, Lyon, 2000

Année 1999

RE'99 "Internet Systematics for Corals". D. Grosser, N. Conruyt, G. Faure, M. Pichon. Reef Encounter, newsletter of the International Society for Reef Studies 26, Allen Press Inc., p. 8, 1999

SFC'99 "Une mesure de dissimilarité pour des données complexes". J. Diatta, D. Grosser, H. Ralambondrainy. Septièmes journées de la Société Francophone de Classification, Nancy, 1999

ICCBR'99 "Managing complex knowledge in natural sciences". N. Conruyt, D. Grosser. Third Int. Conf. on Case-Based Reasoning, K.D. Althoff, R. Bergmann and L.K. Branting (Eds.), LNCS 1650, pp. 401-414, Munich, 1999

ACAI'99 "Tree-based classification approach for dealing with complex knowledge in natural sciences". D. Grosser, N. Conruyt. Machine Learning and Applications, Chania (Greece), pp. 5-16, 1999

NCRI'99 "Development of a Knowledge Base for the Corals of the Mascarene Archipelago". G. Faure, N. Conruyt, M. Pichon, D. Grosser, Y. Geynet. Int. conf. on Scientific Aspects of coral Reef Assessment, Monitoring and Restoration (ISRS/NCRI), Ft. Lauderdale, 1999

Année 1998

ISRS'98 "IKBS : an iterative knowledge base system for managing knowledge in remote systematics. Application to corals of the Mascarene Archipelago" (abstract). N. Conruyt, D. Grosser. Int. Society for Reef Studies, European Meeting, EPHE (Eds.), p. 54, Perpignan, 1998

Année 1996-1997

JOCLAD'97 "Classification et identification d'espèces par discrimination à partir de connaissances structurées". N. Conruyt, D. Grosser, J. Le Renard. Actes des V Jornadas de classificaçao e Analise de dados, pp. 63-66, Porto, 1997

ZNIEFF'97 "Base de connaissances sur les coraux de La Réunion et des Mascareignes". G. Faure, N. Conruyt, M. Pichon, M. Guillaume, D. Grosser, Y. Geynet. Sém. tech. ZNIEFF-mer D.O.M., Guadeloupe, 1997

IFWSDAA'97 "IKBS : An Iterative Knowledge Base System for improving description, classification and identification of biological objects". N. Conruyt, D. Grosser, H. Ralambondrainy. Proceedings of the Indo-French Workshop on Symbolic Data Analysis and its Applications, E. Diday, K.C. Gowda (Eds.), Vol II, pp. 212-224, Paris, 1997

JICAA'97 "Ingénierie des connaissances en Sciences de la vie : application à la systématique des coraux des Mascareignes". N. Conruyt, D. Grosser, G. Faure. Actes des Journées Ingénierie des Connaissances, pp. 513-525, Roscoff, 1997

ICRS'97 "A knowledge base for corals of the Mascarene archipelago : genus Pocillopora" N. Conruyt, G. Faure, G. Ancel, J. Le Renard, M. Guillaume, O. Naïm, N. Gravier-Bonnet, D. Grosser. Proceedings of the 8th Int. Coral Reef Symp., Panama, 1997

PAMM'96 "A Report of a Case Study with Agents in Simulation S. Giroux, P. Marcenac, S. Calderoni D. Grosser, J.R. Grasso". Proc. of the 1st International Conference on Pratical Applications of Intelligent Agents and MultiAgent Technology, London, 1996

Bibliographie

Abelson, H., Sussman, G. J. & Sussman, J. (1989), *Structure et interprétation des programmes informatiques*, InterEditions.

Aimeur, A. (1994), METIS : un système et une Méthode d'Explicitation de Taxinomies destinés à l'Identification de Structures conceptuelles, Université paris vi, Thèse de doctorat.

Arnault, A. & Nicole, P. (1662), *La logique ou l'art de penser*, Flammarion.

Auriol, E. (1995), Intégration d'approches symboliques pour le raisonnement à partir d'exemples. L'induction et le raisonnement par cas dans le diagnostic technique, Thèse de doctorat, Université Paris-IX Dauphine.

Baader, F., Hollunder, B., Nebel, B., Profitlich, H.-J. & Fraconi, E. (1994), 'An empirical analysis of optimization techniques for terminological représentation system', *Journal of Applied Intelligence* **4**(2).

Bailly, N. & Lebbe, J. (1996), 'Disseminating biodiversity information', *Bulletin de la Société Francaise de Systématique* **26**, 4–14.

Baress, E. & Porter, B. (1990), 'Protos : An examplar-based learning apprentice', *Machine Learning : An Intelligent Approach* **3**, chap.4.

Bartlett, F. (1964), *Remembering : a study in experimental and social psychology*, Cambridge University Press.

Beach, J., Pramanik, S. & Beaman, J. (1993), 'Hierarchic taxonomic databases', *Advanced computer methods for systematic biology* pp. 241–256.

Belson, W. (1959), 'Matching and prediction on the principle of biological classification', *Applied Statistics* **8**.

Benzecri, J. (1973), *L'analyse des données*, Dunod.

Bergadano, F. & Gunetti, D. (1995), *Inductive Logic Programming, from Machine Learning to Software Engineering*, MIT Press.

Bisson, G. (1995), 'Why and how to define a similarity measure for object-based representation systems', *Towards very large knowledge bases representation systems (KBKS 95)* pp. 236–246.

Bobrow, D. & Stefik, M. (1988*a*), A common lisp object system specification, SIGPLAN Notices 23, X3J13 Document 88-002R.

Bobrow, D. & Stefik, M. (1988*b*), The loops manual. programming environment, Technical report, Xerox Coporation.

Bobrow, D. & Winograd, T. (1977), 'An overwiew of krl, a knowledge representation language', *Artificial Intelligence* **8**, 155–173.

Boehm, B. (1988), 'A spiral model of software development and enhancement', *IEEE Computer* (21), 61–72.

Borning, A. (1981), 'The programming language aspects of thinglab, a constraint-oriented simulation laboratory', *ACM Transactions on Programming Languages and Systems* **3**(4), 353–387.

Bouquet, A. (1998), 'http : //cdfinfo.in2p3.fr/culture/supernovae/', *source : Web* .

Bourbaki, N. (1974), *Eléments d'Histoire des Mathématiques*, Histoire de la Pensée IV.

Bourgeois, R. (1990), ICEO : Intension, coréférences et objets dans la fédération de formalismes de spécification, Thèse de doctorat, Université Paris VI.

Brachman, R. (1983), 'Krypton : A functional approache to knowledge representation', *COMPUTER* **16**(10), 67–73.

Brachman, R. & Schmolze, J. (1985*a*), 'An overview of the kl-one knowledge representation system', *Cognitive Science* **9**(2), 171–216.

Brachman, R. & Schmolze, J. (1985*b*), 'An overview of the kl-one knowledge representation system', **2**(9), 171–216.

Breiman, L., Friedman, J., Olshen, R. & Schmolze, C. (1984), 'Classification and regression trees', *Wadsworth and Brooks* .

Brito, P. (1991), Analyse des données symboliques, pyramides d'héritages, Thèse de doctorat, Université Paris-IX Dauphine.

Caraux, G. & Lechevallier, Y. (1996), 'Règles de décision de bayes et méthodes statistiques de discrimination', **10**(2), 219–283.

Carré, B. (1989), Méthodologie orientée objet pour la représentation des connaissances. Concepts des points de vue, de représentation multiple et évolutive d'objet, Thèse de doctorat, Université de Lille Flandres Artois.

Carré, B. & Comyn, G. (1988), 'On multiple classification, points of view and object evolution', pp. 49–62.

Carvalho, D. (1991), Méthodes descriptives en Analyse de Données Symboliques, Thèse de doctorat, Université Paris-IX Dauphine.

Carvalho, D. (1998), 'Statistical proximity functions of boolean symbolic objects based on histograms', *Advances in Data Science and Classification* pp. 391–396.

Causse, K. & Lebbe, J. (1995), 'Modélisation des stratégies d'identification par la méthode mcc', *Actes des Journées acquisition, validation, apprentissage (JAVA'95)* pp. 345–358.

Ciampi, A., Diday, E., Lebbe, J., Perinel, E. & Vignes, R. (1996), 'Tree-growing with probabilistically imprecise data', pp. 201–212.

Ciampi, A., Diday, E., Lebbe, J. & Vignes, R. (1994), 'Recursive partition and symbolic data analysis', pp. 277–284.

Codenie, W., Hondt, K. D., Steyaert, P. & Vercammen, A. (1997), 'From custom applications to domain-specific frameworks', *Communications of the ACM* **40**(10), 71–77.

Cointe, P. (1987), Meta-classes are first classes : The objvlisp model, *in* 'Proceedings of the 2nd OOPSLA', ACM, Notices vol.22, n°12, pp. 156–167.

Conruyt, N. (1994), Amélioration de la robustesse des systèmes d'aide à la description, à la classification et à la détermination des objets biologiques, Thèse de doctorat, Université Paris IX-Dauphine.

Conruyt, N. & Grosser, D. (1997), Classification et identification d'espèces par discrimination à partir de connaissances structurées, *in* 'Actes des 5ème journées de classification et d'analyse de données (JOCLAD97)', pp. 63–66.

Conruyt, N. & grosser, D. (1999), Managing complex knowledge in natural sciences, *in* R. B. K.D. Althoff & L. B. (Eds.), eds, 'Third Int. conf. on Case-Based Reasoning (ICCBR'99)', pp. 401–141.

Conruyt, N., Grosser, D., Faure, G., Pichon, M., Geynet, Y., Guillaume, M. & Bouchon, C. (1998), Ikbs : an iterative knowledge base system for managing knowledge in remote systematics. application to corals of the mascarene archipelago, *in* 'Int. Society for Reef Studies (ISRS'98)'.

Coste, H. (1980), *Flore descriptive et illustrée de la France*, Librairie Sciences et Techniques, Albert Blanchard. 2de tirage.

Dallwitz, M. (1974), 'A flexible computer programme for generating diagnostic keys', *Syst. Zoology* **1**(23), 50–57.

Dallwitz, M. (1978), 'User's guide to key. a computer programme for generating identification keys', *CSIRO. Div. Entomol.* **4**, 50–57.

Dallwitz, M. (1984), 'User's guide to the delta system. a general system for coding taxonomic descriptions', *CSIRO. Div. Entomol.* **13**.

Dallwitz, M. (1993), 'Delta and intkey', *Advances in Computer Methods for systematic Biology* .

Dallwitz, M., Pain, T. & Zurcher, E. (1995), 'User's guide to intkey : a program for interactive identification and information retrieval. 1st edition.', *http ://biodiversity.uno.edu/delta/* .

Dallwitz, M., Paine, T. & Zurcher, E. (2000), 'Interactive keys', *http ://biodiversity.uno.edu/delta/* .

Davis, H. (1987), VIEWS, Multiple Perspectives and Structured Objects in Knowledge Representation Language, Bachelor and master of science thesis, MIT.

de Rouen, C. (2001), 'Arborescence des techniques de recherche', *http ://www.chu-rouen.fr/ssf/technfr.html* .

Degoulet, P. (1998), *Informatique médicale*, Abrégés de médecine.

Diatta, J., Grosser, D. & Ralambondrainy, H. (1999), Une mesure de dissimilarité pour des données complexes, *in* '7ème journées de la Société Francophone de Classification (SCF'99)'.

Diday, E. (1971), 'La méthode des nuées dynamiques', *Re. stat. Appliquée* **19**(2), 19–34.

Diday, E. (1987), 'Introduction à l'approche symbolique en analyse de données', pp. 21–56.

Diday, E. (1989), *Introduction à l'approche symbolique en analyse de données*, RAIRO, 23(2).

Diday, E. (1991), 'Des objets de l'analyse des données à ceux de l'analyse des connaissances', *Induction symbolique-numérique à partir de données* .

Diday, E. (1993), 'An introduction to symbolic data analysis'.

Diday, E. & Emilion, R. (1996), *Capacities, Credibilities in Analysis of probabilistic Objects by Histograms and Lattices*, Data Science, Classification and Related Methods, Springer.

Diday, E., Lemaire, J., Pouget, J. & Testu, F. (1982), *Eléments d'analyse des données*, Dunod, Paris.

Drews, O. M. (1993), Raisonnement classificatoire dans une représentation à objets multi-points de vue, Thèse de doctorat, Université Joseph Fourier.

Dubois, D. & Prade, H. (1987), *Théorie des possibilités. Application à la représentation des connaissances en informatique*, Masson.

Ducournau, R. (1988a), Y3. présentation du système, Technical report, Sema Group.

Ducournau, R. (1988b), Yafool. version 3.22. manuel de référence, Technical report, Sema Group.

Ducournau, R., Huchard, M., Libourel, T. & Napoli, A. (1999), 'Aspects classificatoires des systèmes à objets', *Actes des Septièmes Journées de la SFC* pp. 45–52.

Dugertil, P. (1988a), *Contribution à l'étude de la représentation des connaissances fondée sur les objets. Le langage OBJLOG*, Thèse de l'Université d'Aix-Marseille II.

Dugertil, P. (1988b), *OBJLOG II, Guide d'utilisation*, Groupe Représentation et Traitement des Connaissances, Université d'Aix-Marseille II.

Dugertil, P. (1991), 'Inheritance mechanisms in the objlog language : Multiple selective and multiple vertical with points of view in inheritance hierarchies in knowledge representation.', *Wiley and Sons Ltd.* pp. 245–256.

Euler, L. (1747), *Opera Omnia*, 46 vol. paus, vol. 2.

Euzenat, J. (1998), 'Représentation de connaissance par objets', pp. 293–319.

Euzenat, J. (1999), Représentation des connaissances : de l'approximation à la confrontation, H.d.r., Université Joseph Fourier.

Fahlman, S. (1979), 'Netl : A system for representing and using real world knowledge'.

Faure, G. (1982), Recherche sur le peuplements de Scléractiniaires des récifs coralliens de l'Archipel des Mascareignes, Thèse de doctorat d'etat, Université d'Aix-Marseille II.

Faure, G., Conruyt, N., Pichon, M., Guillaume, M., Grosser, D. & Geynet, Y. (1999), 'Development of a knowledge base for the corals of the mascarene archipelago', *Int. Conf. on Scientific Aspects of Coral Reef Assessment, Monitoring and Restoration* .

Ferber, J. (1989), Objets et agents : une étude des structures de représentation et de communication en Intelligence Artificielle, Thèse de doctorat, Université Paris-VI.

Ferber, J. (1990), 'Ptitloo, petit langage objet', *http ://www.lirmm.fr/ ferber/* .

Fisher, D. & Langley, P. (1985), 'Approaches to conceptual clustering', *Proc. of IJCAI* pp. 691–697.

Fox, M., Wright, J. & Adam, D. (1986), 'Experiences with srl : An analysis of a frame-based knowledge representation', *Expert Database Systems* pp. 161–172.

Frege, G. (1893), *Grundsetze des Arithmetik, begriffsschriftlich abgeleitet*, Vol. 1, dans Ecrits logiques et philosophiques.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995), *Design Patterns : Elements of Reusable Object Oriented Software*, Addison-Wesley.

Garey, M. & Johnson, D. (1979), 'The need of biases in learning generalisations', *Computers and Intractability : A Guide to the Theory of NP-Completeness* .

Girard, R. (1997), Classification Conceptuelle sur des Données Arborescentes et Imprécises, Thèse de doctorat, Université de la Réunion.

Goldberg, A. & Robson, D. (1983), *Smalltalk-80, The Language and its implementation*, Addison-Wesley.

Gowda, K. C. & Diday, E. (1991), 'Symbolic clustering using a new dissimilarité measure', *Pattern Recognition* **24**, 567–578.

Grosser, D. & Conruyt, N. (1999), Tree-based classification approach for dealing with complex knowledge in natural sciences, *in* 'ACAI'99, Machine Learning and Applications'.

Guarino, N. (1998), 'Formal ontology in information systems', *IOS Press* .

Hall, A. (1970), 'A computer-based system for forming identification keys', *Taxon* **19**.

Hendrix, G. (1979), 'Encoding knowledge in partitioned networks', *Associative Networks : The Representation and Use of Knowledge by Machine* .

Hunt, E., Marin, J. & Stone, P. (1966), *Experiments in induction*, dans Ecrits logiques et philosophiques.

Ichino, M. (1986), 'General metrics for mixed features : the cartesian space theory for pattern recognition', *IEEE Intl Conf. Syst. Man Cybern.* pp. 14–17.

Ichino, M. (1988), 'Pattern classification based on the cartesian join system : a general tool for feature selection', *IEEE Intl Conf. Syst. Man Cybern.* .

Ichino, M. (1994), 'Feature selection for symbolic data classification', *New Approches in Classification and Data Analysis* pp. 423–429.

Jaczynski, M. (1998), Modèle et plate-forme à objets pour l'indexation des cas par situations comportementales : application à l'assistance à la navigation sur le web, Thèse de doctorat, Université de Nice-Sophia Antipolis.

Jambu, M. (1999), *Méthodes de base de l'analyse de données*, Eyrolles.

Johnson, R. E. & Foote, B. (1988), 'Designing reusable classes', *Object-oriented Programming* **1**(2), 22–35.

Kayser, D. (1984), 'Examen des diverses méthodes utilisées en représentation des connaissances', *Actes du 4ème CARFIA* pp. 115–144.

Keene, S. E. (1989), *Object-Oriented Programming in Common LISP : A Programmer's Guide to CLOS*, Reading : Addison-Wesley.

Khinchin, A. (1957), *Mathematical foundations of information theory*, Dover books on intermediate and advanced mathematics.

Kim, W., Banerjee, J., Chou, H., Garza, J. & Woelk, D. (1987), Composite object support in an object-oriented database system, *in* 'Proceedings of the 2nd OOPSLA', ACM, Notices vol.22, n°12, pp. 118–125.

Kodratoff, Y. (1991), 'Faut-il choisir entre science des explications et science des nombres ?', *Induction symbolique et numérique à partir des données* **3**.

Krief, J. (1990), Un système interactif de construction d'environnements de prototypage de multiples outils d'interprétation de modèles de représentation, Thèse de doctorat, Université Pierre et Marie Curie (Paris 6).

Kuhn, T. (1983), *La structure de la révolution scientifique*, Flammarion.

Lambert, J. M. & Williams, W. (1966), 'Multivariate methods in plant ecology', *J. Eco.* **54**.

Langefors, B. (1977), 'Information systems theory', *Inf Syst. 1977* **2**, 207–219.

Lavrac, N. & S.Dzeroski (1994), *Inductive Logic Programming*, Ellis Horwood.

Lebbe, J. (1991), Représentation des concepts en biologie et en médecine. Introduction à l'identification assistée par ordinateur, Thèse de doctorat, Université Paris-VI.

Lebbe, J. (1995), 'Notice individuelle pour le recrutement à un emploi de professeur des université', *cité dans (Vignes, 2000)* p. 152 pages.

Lebbe, J. (1998), 'Représentations par objets et classifications biologiques', *[Ducournau 98]* pp. 421–447.

Lebbe, J. & Vignes, R. (1993), 'Makey 3.0, user manual', *Machine Learning Toolbox* .

Levesque, H. (1986), 'Knowledge representation and reasoning', *Annual Reviews of Computer Science* pp. 255–287.

Levesque, H. & Mylopoulos, J. (1979), 'A procedural semantics for semantic networks', *Associative Networks : The Representation and Use of Knowledge by Machine* .

Lévi, C., Potier, P. & Sévenet, T. (1978), 'A la découverte', *Courrier du C.N.R.S.* **29**, 47–51.

Lieberman, H. (1987), *A Preview of ACT 1*, AI Memo 625, AI Lab., MIT.

Manago, M. (1991), 'Kate : intégration de techniques numériques et symboliques en apprentissage', *Induction Symbolique et Numérique à Partir de Données* .

Mariño, O., Rechenmann, F. & Uvietta, P. (1990), Multiple perspectives and classification mechanism in object-oriented representation, *in* 'Actes du 9th ECAI', pp. 425–430.

Masini, G., Napoli, A., Colent, D., Léonard, D. & Tombre, K. (1989), *Les langages à objets*, InterEditions.

Matile, L., Tassy, P. & Goujet, D. (1987), 'Introduction à la systématique zoologique', *Biosystema 1* .

Merz & Murphy (1996), 'Uci repository of machine learning databases', *Department of Information and Computer Science* .

Meyer, B. (1990), *Conception et programmation par objets, pour du logiciel de qualité*, InterEdition.

Michalski, R. (1986), 'A theory and methodologie of inductive learning', *Machine Learning : An Artificial Intelligence Approach* **1**, 83–129.

Michalski, R. S. (1983), 'A theory and methodology of inductive learning', *Machine Learning : An Artificial Intelligence Approach* **1**, 83–134.

Microsystem, S. (2001), 'java.sun.com'.

Mingers, J. (1989), 'An empirical comparison of selection measures for decision tree induction', *Machine Learning* **3**(4), 227–243.

Minsky, M. (1975), 'A framework for representing knowledge', *The psychologie of Computer Vision* .

Mitchell, T. M. (1990*a*), 'Generalisation as search', *Readings in Machine Learning* pp. 96–107.

Mitchell, T. M. (1990*b*), 'The need of biases in learning generalisations', *readings in Machine Learning* pp. 184–191.

Mitchell, T. M. (1997), *Machine Learning*, McGraw-Hill.

Moon, D. (1986), 'Object-oriented programming with flavors', *Proceedings of the 1st OOPSLA* pp. 1–8.

Muggleton, S. (1992), 'Inverting implication', *Proc. of the Second Int. Workshop on Inductive Logic Programming (ILP92)* .

Napoli, A. (1992), Représentation à objets et raisonnement par classification en intelligence artificielle, Thèse de doctorat, Université Université Nancy 1.

Napoli, A. (1994), 'Catégorisation, raisonnement par classification et raisonnement à partir de cas', *Journées Acquisition, Validation et Apprentissage (JAVA'94)* pp. 1–14.

Napoli, A. (1997), Une brève introduction aux logiques terminologiques, Rapport technique 3314, INRIA.

Nebel, B. (1990), 'Reasoning and revision in hybrid representation systems', *Lecture Notes in Computer Science* **422**.

Niquil, Y. (1993), Acquisition d'exemples en discrimination, Thèse de doctorat, université paris ix, Dauphine.

Nogier, J. (1991), *Génération automatique de langage et graphes conceptuels*, Editions HERMES.

O.J.Dahl & Nygaard, K. (1966), 'An algol-based simulation language', *ACM* **9**(9), 671–678.

Pachet, F. (1997), Représentation de connaissances et langages à objets, H.d.r., Université Pierre et Marie Curie (Paris 6).

Pankhurst, R. (1970), 'A computer program for generating diagnostic keys', *Computer Journal* .

Pankhurst, R. (1986), 'A package of computer programs for handling taxonomic databases', *Computer Applications in the Biosciences* **2**(1), 33–39.

Pankhurst, R. (1991), *Practical Taxonomy Computing*, Cambridge University Press.

Pankhurst, R. (1998), 'A historical review of identification with computers', *Information Technology Plant Pathology and Biodiversity* pp. 229–240.

Payne, R. (1975), 'Genkey : a programme for constructing diagnostic keys', *Biological Identification with Computers* **4**, 65–72.

Peirce, C. (1965), 'Elements of logic', *Collected Papers of Charles Sanders Peirce (1839-1914)* **2**.

Perinel, E. (1996), Segmentation et analyse de données symboliques : application à des données probabilistes imprécises, PhD thesis, Université Paris IX-Dauphine.

Perrot, J. & Napoli, A. (1996), 'Systèmes à objets : tendances actuelles et évolution', **15**(6).

Planche, R. (1988), *Maîtriser la modélisation conceptuelle*, Editions Masson.

Plotkin, G. (1970), 'A note on inductive generalization', *Machine Intelligence* **5**, 153–163.

Polaillon, G. (1998), Organisation et interprétation par les treillis de Galois de données de type multivalué, intervalle ou histogramme, Thèse de doctorat, Université Parix IX-Dauphine.

Pólya, G. (1958), *Les mathématiques et le raisonnement plausible*, Gauthier-Villars.

Popper, K. (1973), *La logique de la découverte scientifique*, Payots (Eds) Press.

Popper, K. R. (1978), *La connaissance objective*, Complexe (Eds.).

Puerta, A. R., Egar, J. W., Tu, S. W. & Musen, M. A. (1992), 'A multiple-method knowledge acquisition for the automatic generation of knowledge acquisition tools', *Knowledge Acquisition* **4**, 171–196.

Quillian, M. (1968), 'Semantic memory, in semantic information processing', *Semantic Information Processing* pp. 227–270.

Quinlan, J. (1979), 'Discovering rules from large collections of examples : a case study', *Expert Systems in the micro electronic age* pp. 691–697.

Quinlan, J. (1986), 'Induction of decision trees', *Machine learning* **1**, 81–106.

Quinlan, J. (1992), *C4.5. Programs for Machine Learning*, Morgan Kaufman.

Quinlan, J. R. (1996), 'Improved use of continuous attributes in c4.5', *Journal of Artificial Intelligence Research* **4**, 77–90.

Quinlan, J. R. & Cameron-Jones, R. M. (1993), 'Foil : A midterm report', *ECML'93* **667**, 3–20.

Raphael, B. (1968), 'Sir : A computer program for semantic information retrieval', *Semantic Information Processing* pp. 33–135.

Rechenmann, F. (1985), Shirka : mécanismes d'inférences sur une base de connaissances centrée objets, *in* 'Actes du 5ème CARFIA', Grenoble, France, pp. 1243–1254.

Rechenmann, F. (1988), Shirka : système de gestion de bases de connaissances centrées-objet. manuel de référence, Technical report, INRIA/ARTEMIS.

Rechenmann, F., Fontanille, P. & Uvietta, P. (1990), Shirka : système de gestion de bases de connaissances centrées-objets, Rapport de recherche, INRIA et Laboratoire Artémis / IMAG.

Reichgelt, H., Jackson, P. & Harmelen, F. V. (1989), *Logic-based knowledge representation*, MIT Press.

Renard, J. L., Lévi, C., Conruyt, N. & Manago, M. (1996), 'Sur la représentation et le traitement des connaissances descriptives : une application au domaine des éponges du genre hyalonema', **66**, 37–48.

Roberts, R. & Goldstein, I. (1977), *The FRL Manual*, AI Memo 409, MIT, Cambridge, Massachusetts.

Roche, M., Aubert, S., Brand, P., Choler, P., Cunzi, M. & Douzet, R. (2000), 'La station alpine du lautaret', *http ://www.ujf-grenoble.fr/JAL* .

Rosch, E. & Lloyd, B. (1978), 'Cognition and categorization', *Principles of categorization* pp. 27–48.

Rosch, E., Mervis, C., Gray, W., Johnson, D. & Boyes-Bream, P. (1976), *Basic Objects in Natural Categories*, Vol. 8, Cognitive Psychology.

Rosnay, J. D. (1975), *Le Macroscope*, Editions du Seuil.

Rousse, G. (2001), 'Osis, an object model for systematics', *TDWG 2001* .

Rousse, G., Grosser, D., Conruyt, N. & Vignes, R. (2000), 'Object-oriented systematics information system (osis)', *International Conference on Coral Reef Society (ICRS 2000)* .

Schmid, H. A. (1997), 'Systematic framework design by generalization', *Communications of the ACM* **40**(10), 48–51.

Shapiro, E. Y. (1981), 'An algorithm that infers theory from facts', *Tech Report 192* .

SHERPA, P. (1995), Tropes 1.0 reference manual, Rapport de recherche, INRIA Rhône-Alpes.

Shreiber, G., Wielinga, B., de Hoog, R., Akkermans, H. & de Velde, W. V. (1994), 'Commonkads : a comprehension methodology for kbs development', *IEEE Computer* (9), 8–37.

Simon, A. (1995), Une approche de la fouille de données avec application au domaine médical., Mémoire de dea, Université Université Nancy 1.

Simon, A. (2000), Outils classificatoires par objets pour l'extraction de connaissances dans des bases de données, Thèse de doctorat, Université Université Nancy 1.

Simon, A. & Napoli, A. (1997), 'Un algorithme de fouille dans une représentation des données par objets : une application au domaine médical', *Journées Ingénierie des Connaissances et Apprentissage Automatique (JICAA'97)* (4), 593–604.

Sneath, E. & Sokal, E. (1973), *Numerical Taxonomy*, W.H. Freeman, San Francisco.

Soune-Seyne, N. & Cadet, J. (2001), Classification par hiérarchie faible conceptuelle. projet de maîtrise d'informatique encadré par j. diatta, d. grosser et h. ralambondrainy, Technical report, IREMIA, Université de la Réunion.

Sowa, J. (1984), 'Conceptual structures', *Information Processing in Mind and Machine* .

Sowa, J. (1991), 'Toward the expressive power of natural language', *Principles of Semantic Networks : Exploration in the Representation of Knowledge* p. chap. 5.

Stepp, R. & Michalski, R. (1986), 'Conceptual clustering : Inventing goal-oriented classifications of structured objects', *Machine Learning : An Artificial Intelligence Approach* **2**, 471–498.

Stroustrup, B. (1989), *Le Langage C++*, InterEdition.

Sussman, G. & Steele, G. J. (1980), 'Constraints-a language for expressing almost-hierarchical descriptions', *Artificial Intelligence* **14**, 1–39.

Sutcliffe, J. P. (1993), 'Concept, classe, and category in the tradition of aristotle', *Dans Categories and Concepts Theorical Views and Inductive Data Analysis* pp. 35–65.

Tardieu, H., Rochfeld, A. & Coletti, R. (1984), *La méthode MERISE*, Les éditions d'organisation.

Tayse, A., Delsarte, P., Hagelstein, J., Louis, G., Rifaut, A., Vauclair, M., Dubois, E., Lamsweerde, A. V. & Linden, F. V. D. (1990), *Approche Logique de l'Intelligence Artificielle*, Vol. 1, Dunod.

Tsichritzis, D. & Lochovsky, F. (1982), *Data Models*, Prentice-Hall.

Valtchev, P. (1999), Construction automatique de taxonomies pour l'aide à la représentation de connaissances par objets, Thèse de doctorat, Université Grenoble 1.

Valtchev, P. & Euzenat, J. (1997), 'Dissimilarity measure for collections of objects and values', *Lectures notes in computer science* **1280**, 259–272.

Ver (n.d.). www.versant.com.

Veron, J. (2000), *Corals of the world*, Australian Institue of Marine Science.

Veron, J., Pichon, M. & Wisjman-Best, M. (1976), 'Scleractinia of eastern australia. families thamnasteriidae, astrocoeniidae, pocilloporidae', **1**(3), 1–233.

Vidot, N. (1999), Approche multi-agents pour la classification automatique. projet de maîtrise d'informatique encadré par h. ralambondrainy, Technical report, IREMIA, Université de la Réunion.

Vignes, R. (1991), Caractérisation automatique de groupes biologiques, Thèse de doctorat, Université Paris-VI.

Vignes, R. (1996), 'De la construction automatique de clés d'identification aux stratégies d'identification', *Biosystema 14* pp. 91–108.

Vignes, R. (2000), Informatique et Systématique : vers une ontologie de la Systématique, H.d.r., Université Paris-VI.

Voyer, R. (1989), Implémentation d'architectures efficaces pour la représentation des connaissances, Thèse de doctorat, Université Pierre et Marie Curie (Paris 6).

Wielenga, B., Screiber, A. & Breiker, J. (1992), 'Kads : a modelling approach to knowledge engineering', *Knowledge Acquisition* **4**.

Wilson, D. & Martinez, T. (1997), 'Improved heterogeneous distance functions', **6**, 1–34.

Winograd, T. (1972), *Understading Natural Language*, New York, Acdemic Press.

Winston, M., Chaffin, R. & Herrmann, D. (1987), 'A taxonomy of part-whole relations', *Cogniive Science* **11**, 417–444.

Winston, P. (1977), *Artificial Intelligence*, Addison Wesley, Reading.

Wolinski, F. (1990), Etude des capacités de modélisation systémique des langages à objets appliqués à la représentation de robots, Thèse de doctorat, Université Pierre et Marie Curie (Paris 6).

Woods, W. (1975), 'What's in a link : Foundations for semantic networks, in representation and understanding', *Representation and Understanding* pp. 33–135.

Zadeh, L. A. (1965), 'Fuzzy sets', *Information and control* **8**, 338–353.

Résumé

La modélisation du savoir-faire des systématiciens à l'aide de Bases de Connaissances, offre le moyen de mieux comprendre, préserver et transmettre aux générations futures les connaissances sur la biodiversité des espèces. Pour construire une base de connaissances, il est nécessaire de disposer de méthodes de gestion de connaissances, de modèles de représentation et d'outils informatiques adaptés d'une part, à la complexité des concepts manipulés par les systématiciens et d'autre part, à la richesse des descriptions des spécimens représentatifs des espèces. La plate-forme logicielle proposée, appelée $\mathcal{IKBS}$, offre aux Systématiciens un environnement pour la construction de Bases de Connaissances évolutives, ainsi qu'une aide à l'identification et à la classification d'objets complexes. Elle met en oeuvre une méthodologie itérative fondée sur l'approche expérimentale de nature inductive des naturalistes. Le paradigme objet utilisé pour sa réalisation favorise l'extensibilité et la réutilisabilité des composants logiciels développés. $\mathcal{IKBS}$ associé aux technologies multimédia et hypertextuelles a été utilisé par une groupe d'experts pour la construction d'une « base de connaissances sur les coraux des Mascareignes ».

Mots-clés: Représentation des Connaissances, Bases de Connaissances, Identification et Classification d'objets complexes, Systématique des Coraux, Plate-forme à objets, Java.

Abstract

The ability to model taxonomic information using Knowledge Bases allows a better understanding, the preservation and the transmission to future generations of biological knowledge. There are three main elements that intervene in the building of a Knowledge Base : methods for managing knowledge, models for representing knowledge and appropriate software tools that are able to deal with the complexity of the concepts and the richness of the descriptions manipulated by systematicians. The proposed software environment, named $\mathcal{IKBS}$, not only enables systematicians to build evolving Knowledge Bases but also offers tools for identifying and classifying complex descriptions. It uses an iterative methodology based on the experimental and inductive approach usually adopted by naturalists. The object paradigm used for its implementation facilitates the extensibility and the reusability of developed software components. The software environment $\mathcal{IKBS}$, combined with multimedia and hypertext technologies, was used by a group of experts for building a knowledge base on Mascareignes Corals.

Keywords: Knowledge Representation, Knowledge Bases, Identification and Classification of complex objects, Corals Systematics, object-oriented framework, Java.