Quelques exemples d'élaboration de modèles conceptuels

Roland Billon, Médiaconstruct et Enseignant à l'ENSA de Marseille
Isabelle Fasse, Médiaconstruct, ex Enseignante à l'ENSA de Marseille
Jacques Zoller, Professeur à l'ENSA de Marseille
Pascal Tonarelli, Maître de conférences à l'UVHC
Hafida Boulekbache, Maître de conférences à l'UVHC
Stéphane Duriez, médiatiseur à l'UVHC

Table des matières

Chapitre I. Résumé et prise de conscience 5

Chapitre II. Savoir lire UML, connaissant NIAM 6

A. Des différences de détails pour le même aspect (statique) 6

Chapitre III. Un diagramme UML pour chaque point de vue 12

A. Le modèle dynamique dans tous ses états ! 12
B. Le diagramme “Scénario” 14
C. Le diagramme “d'Etats” 16
D. Sous-état, concurrence et synchronisation d'états 19
E. Le diagramme de “Collaboration” 20
F. Ne pas se tromper de cible : faire porter l'information par les acteurs ou par les objets ? 23

Chapitre IV. Pour approfondir : les diagrammes conceptuels des IFC exprimés en EXPRESS-G 25

A. Le formalisme de spécification formelle Express-G 25
B. La circulation entre les pages d'un schéma Express-G 28
C. Début du parcours d'un schéma Express-G des IFC : l'arbre général 30
D. ... Puis examinons la page 4 du Schéma IfcProductExtension 32
E. ... Puis la page 2 du Schéma IfcProductExtension 35
F. ... Puis les autres pages du Schéma IfcProductExtension 38

ChapitreI. Résumé et prise de conscience

Résumé :

Deux exemples sont abordés : La méthode de spécification formelle *UML*, l'une des plus utilisées dans le monde, qui s'étend aux trois aspects fondamentaux d'un modèle formel : statique, dynamique et fonctionnel.

Nous en établissons une comparaison avec la méthode *NIAM*.

Puis, pour les curieux aussi bien que ceux qui sont intéressés par les *systèmes d'information* dans la Construction, nous pénétrons dans le cœur du modèle IFC, en suivant le guide des relations d'héritage.

Prise de conscience :

L'étudiant architecte a tendance à oublier la prise en compte du temps dans un modèle de bâtiment, alors que l'étudiant ingénieur y est préparé par les procédures de calcul et de gestion de chantier qui sont essentiellement contraintes par l'aspect dynamique.

Mais tous seront conscients que les outils informatiques du secteur ne peuvent être développés sans une coopération entre des professionnels des métiers de la Construction et des informaticiens à travers les modèles conceptuels.

Les Unités "Se comprendre autour d'un système d'information" et "Quelques exemples d'élaboration de modèles conceptuels" de ce cours se limitent à un survol, une prise de conscience des méthodes à mettre en oeuvre.

Selon votre objectif professionnel, vous devrez approfondir vous même.

ChapitreII. Savoir lire UML, connaissant NIAM

A. Des différences de détails pour le même aspect (statique).

Ce chapitre est pour vous, cher lecteur, l'occasion de mettre en pratique une des conclusions de la précédente unité de cours : votre capacité à vous adapter à toute méthode de spécification formelle autre que NIAM.

Nous comparons rapidement les syntaxes des formalismes NIAM et *UML* dans le tableau suivant. Pour information, la collection des symboles, et leur définition sémantique que nous examinons se nomment dans le jargon des modèles conceptuels « le *méta-modèle* ». Par ailleurs l'examen de la méthode *UML* est un prétexte pour aborder de nouveaux concepts.

D'ou vient la méthode *UML* (Unified Modeling Language) ?

S'appuyant sur les concepts objets, c'est une synthèse entre trois méthodes assez voisines très répandues : *OMT* (Object Modeling Technique), **Booch**, et *OOSE* (Object Oriented Software Engineering). Les trois auteurs de ces méthodes ont coopéré à cette synthèse UML :

Rumbaugh James, *Booch Grady*, *Jacobson Ivar*.

Synthèse réussie, à constater la rapidité avec laquelle *UML* se répand, et la liste des utilitaires logiciels qui intègrent ses diagrammes dans les ateliers de génie logiciel.

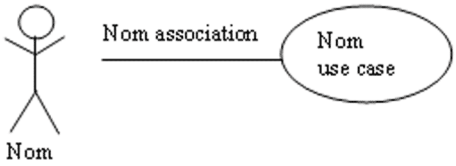
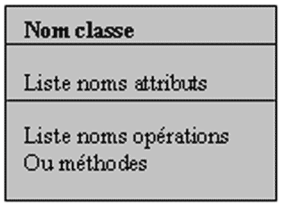
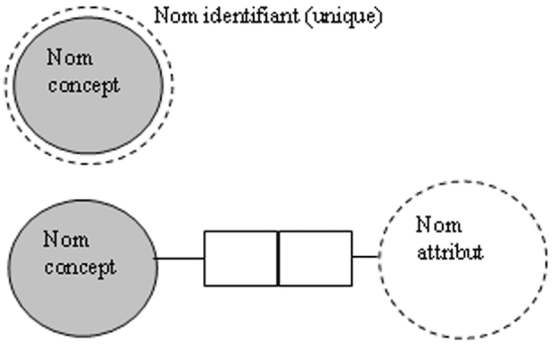
On constate que NIAM est à un niveau de *granulométrie* plus fin, qui tient à un découpage binaire de l'aspect relationnel. Il oblige à une analyse des idées décomposées jusqu'à un niveau réellement élémentaire. Il permet une correspondance avec les propriétés de l'algèbre ensembliste. De ce point de vue, NIAM offre un avantage pédagogique certain.

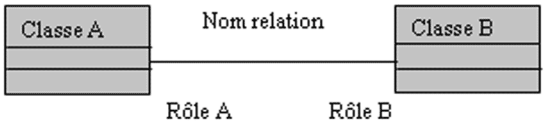
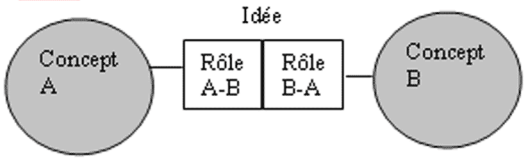
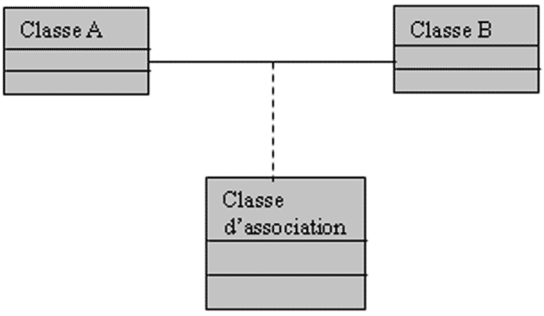
En revanche, il est moins pratique dans l'élaboration opérationnelle des modèles. Il est moins concis qu'UML. NIAM a les défauts de ses qualités.

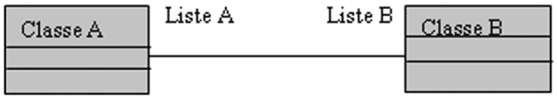
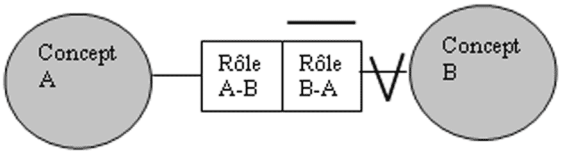
On a donc raison de préférer UML dans un contexte professionnel, qui permet beaucoup plus de libertés, par exemple dans l'expression des cardinalités, et qui de plus est plus complet, proposant également un modèle dynamique. Mais il est plus dangereux, car les incohérences sont plus difficilement décelables.

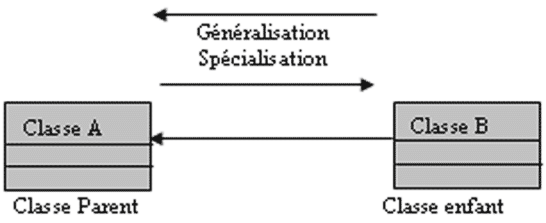
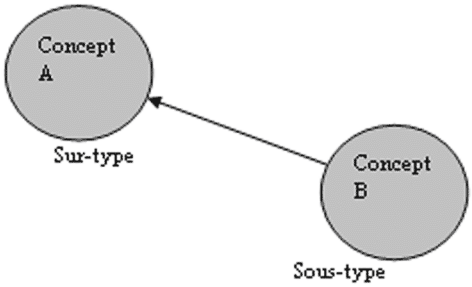
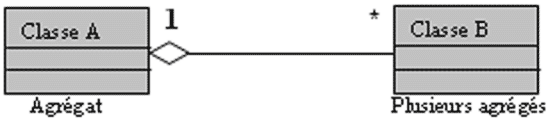
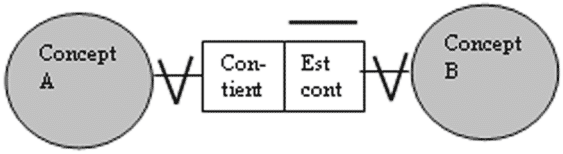
Vous trouverez à l'extérieur de cette unité, en fichier complémentaire, un fichier PDF "UML_NIAM.pdf" des différences de détails entre UML et NIAM pour un même aspect.

Le contenu de ce fichier peut être agrandi et imprimé pour vous permettre de mieux visualiser ces informations que ne le permettent les images à l'intérieur de cette page web.

UML : Modèle statique	**NIAM**
Acteur, use case, **association** 	concepts absents
Classe, attribut, **opération** 	Concept **ou ensemble**, **identifiant** 
L'identifiant est absent. Il est repéré par un attribut à qui on donne implicitement le statut de principal. A n'en pas douter, l'évolution d'UML comblera cette absence.	Concept opération absent

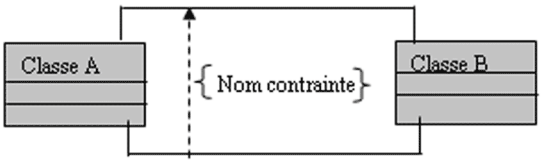
UML : modèle statique	NIAM
Association (entre deux classes) Rôles  	Idée (ou relation binaire) entre deux concepts (ensembles) Rôles 
Classe d'association  	Ce concept absent de NIAM est très pratique pour qualifier le nouvel ensemble constitué par les couples d'objets pris dans l'association A-B

UML : modèle statique	NIAM
Cardinalités : ensemble des entiers positifs  Exemples = 0..1 ; 1 ; 1..3 ; * * = plusieurs, On peut de plus exprimer des contraintes de cardinalité sur chaque ensemble. (Voir plus loin) Les cardinalités du sens de l'association B vers A sont obligatoirement l'inverse de celles de A vers B.	**Cardinalités seulement binaires**  Absence de symbole : pas de contrainte Unicité Totalité Les cardinalités du sens de l'association B vers A sont indépendantes de celles de A vers B .

UML : modèle statique	NIAM
Généralisation-spécialisation  	**Relation d'héritage** 
Classe abstraite (sans instances)	**Abstract super-type (sans occurrences)**
Association particulière d'agrégation (composé de, composant ou faisant partie de)  Le symbole flèche « losange » se place toujours du coté de l'agrégat, le contenant. Traduction en phrase élémentaires : Un objet de la classe A , par exemple un Appartement, est composé de plusieurs pièces. Plusieurs pièces font partie d'un appartement. La sémantique de cette association implique deux propriétés - les associations opèrent une partition exclusive dans l'ensemble agrégé (une pièce ne peut appartenir à deux appartements) - si on supprime un appartement, les pièces associées disparaissent. Pour conserver explicitement la trace du contenu de chaque Appartement, il faudrait créer une classe d'association.	NIAM ne distingue pas de noms particuliers pour une idée. Ce sont les rôles qui définissent la sémantique de la relation. Cette relation d'agrégation se traduit par le schéma : 

UML : modèle statique

Contraintes entre deux associations

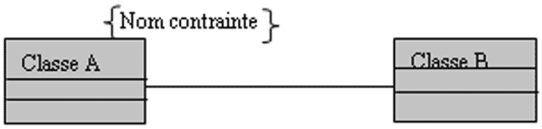


Se note entre accolades.

Contrairement à NIAM , les contraintes, leur nom, leur signification dans UML sont libres.

Attention de ne pas en abuser : l'informaticien devra les traiter !

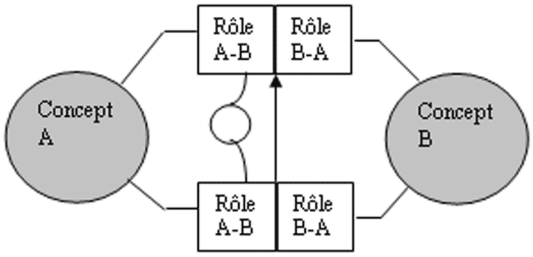
Contrainte portant sur les instances d'une classe :

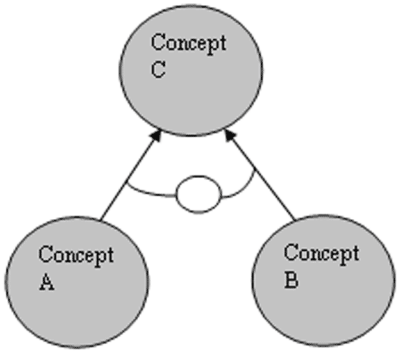


Le principe de qualification libre d'une contrainte sur les occurrences d'un concept n'existe pas dans NIAM

NIAM

Contraintes entre deux idées ou entre deux relations d'héritage

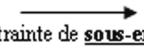




Le symbole

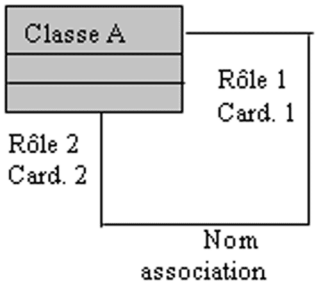
Peut contenir une contrainte

- **d'unicité**
- **d'égalité**
- **d'exclusion**
- **de totalité**

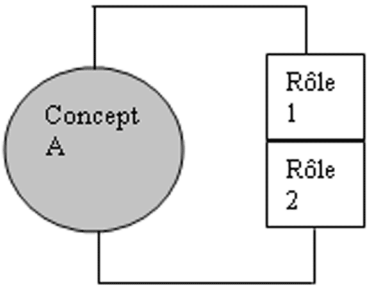
Le symbole 

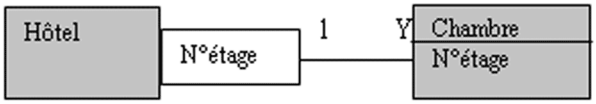
Est une contrainte de **sous-ensemble**.

Association unaire



Idée portant sur le même concept



UML : modèle statique	NIAM
Les qualificateurs. Prenons un exemple  Traduction : Dans un hôtel, il y a des chambres, qui portent entre-autre un numéro d'étage comme attribut. Le qualificateur permet de compter les chambres par étage : Pour un numéro d'étage, il y a Y chambres.	Dans NIAM il vaut mieux créer un concept à la place d'un attribut chaque fois que l'on doit utiliser un identifiant comme ici le numéro. Le schéma NIAM correspondant à cet exemple comportera donc 3 concepts (Hôtel, Chambre et étage), et deux idées. Le schéma est moins concis, mais gagne en clarté. C'est d'ailleurs vrai aussi pour UML.

Voici une liste de définitions de termes utilisés dans les images ci dessus :
Acteur
Use case
Classe
Attribut
Concept
Rôles
Idée
Unicité
Totalité
Héritage
Contraintes

ChapitreIII. Un diagramme UML pour chaque point de vue

A. Le modèle dynamique dans tous ses états !

Le *modèle dynamique* s'appuie sur les éléments du modèle statique que nous venons de passer en revue. Il le complète en y ajoutant des concepts destinés à formaliser le fonctionnement du *systèmes d'information* dans le temps :

- *Evénement* et *messages* reçus ou émis par une classe, ou par un *Acteur*
- *Etat* des occurrences d'une classe en distinguant *Etat* **initial** et l'*Etat* **final**
- *Transition*, qui formalise un changement d'état
- *Attribut* et *gardiens* d'une *Transition*
- *Opération* ou méthode, en distinguant l'*activité* et l'*action*
- La *concurrence* et la *synchronisation* d'états.

Ces 12 nouveaux concepts s'expriment dans des types de diagrammes distincts :

- Le *diagramme Scénario*
- Le *diagramme d'états*
- Le *diagramme de collaboration*

On peut envisager pour la description d'une base de données, de réunir tous les schémas statiques du modèle en un seul gigantesque schéma qui représente la totalité du modèle.

Un formalisme permet d'ailleurs de le scinder en parties plus lisibles, et de circuler entre ces différents sous-schémas, tout en marquant les références vers les autres (**Les packages**).

Cela n'a pas de sens pour les diagrammes dynamiques.

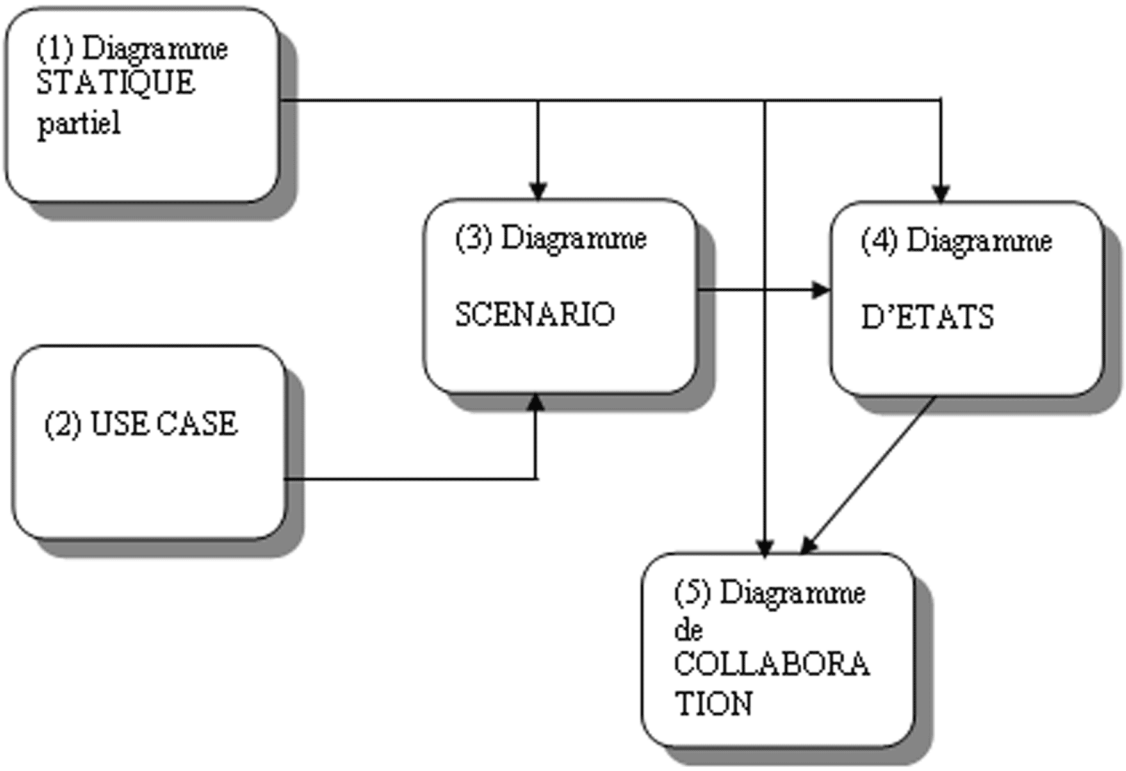
Leur rôle est limité à la description séparée des petites et grandes fonctions du modèle. Participent donc à un diagramme dynamique uniquement les classes concernées, avec bien sûr leurs entités associées.

Pour donner un ordre de grandeur, il peut exister dans un modèle UML autant de diagrammes dynamiques (*diagramme de collaboration*) qu'il existe de *Use case* un peu complexe.

Donc pour un logiciel de CAO, souvent plus d'une centaine.

En général, la méthode d'élaboration d'un diagramme dynamique est la suivante :

(Légende : chaque rectangle arrondi représente un type de diagramme, une flèche une reprise d'informations qui marque la *Traçabilité* des entités du modèle. L'ordre chronologique logique est repéré par un numéro)


(1) Diagramme STATIQUE partiel
(3) Diagramme SCENARIO
(4) Diagramme D'ETATS
(2) USE CASE
(5) Diagramme de COLLABORATION

Dans quelle mesure vous, professionnel (utilisateur, *Acteur*) non informaticien, devez participer à l'élaboration de ces diagrammes dynamiques ?

Pour plusieurs raisons :

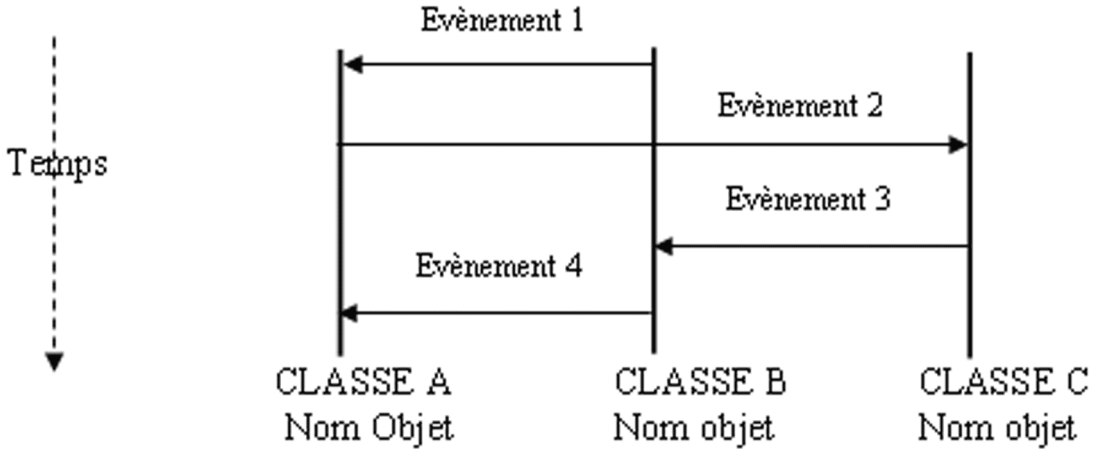
- L'étude du déroulement dynamique (et donc ergonomique) d'une fonction provoque le plus souvent des compléments dans le modèle statique, surtout au niveau de l'implémentation des *Attribut* à renseigner dans la base de données. Cette étude précise la sémantique du modèle, qui relève de votre domaine de compétence.
- Votre participation en tant que responsable du métier est indispensable pour affiner les aspects fonctionnels du modèle et bien vérifier qu'ils seront pris en compte par les informaticiens.
- Une réflexion sur le déroulement des fonctions permet d'optimiser l'ergonomie du système, la constitution de ses boîtes de dialogue, et l'intégration d'une démarche d'analyse de la valeur.
- Enfin, à cette occasion, vous établissez naturellement un guide pour les tests, l'évaluation des performances et la validation du système une fois développé. Il vous suffira de faire fonctionner le système pour chaque diagramme étudié, lequel est directement issu des *Use case* que vous avez imaginé.
- Par ailleurs, vous allez vous apercevoir que l'utilisation de ces diagrammes dynamiques, par leur portée générale, dépasse le cadre strict de l'élaboration d'un modèle d'un système d'information. Vous pourrez les utiliser pour mettre au point n'importe quelle méthode organisationnelle, ou problème de planning faisant intervenir des objets, des fonctions, et le temps. C'est à dire presque tous les évènements de la vie professionnelle ou même courante !

Passons en revue chaque diagramme, et les nouveaux concepts utilisés.

B. Le diagramme “Scénario”

Un *diagramme Scénario* est constitué d'*Evénement* mettant en jeu des objets de plusieurs classes, et cités dans un ordre chronologique. Son objectif n'est pas de représenter le réel d'une façon exhaustive. C'est un document d'analyse pour mieux cerner l'enchaînement des *Evénement* d'une fonction qui a été par exemple esquissée dans un *Use case*.

On le représente par une sorte de diagramme à barres :



Les concepts de *Classe* et d'*objets* sont ceux du modèle statique, et les objets en proviennent. Ils sont représentés par une barre verticale en trait épais.

Un nouveau concept et une spécialisation de ce concept interviennent :

Evènement et message.

Comme son nom l'indique, l'*Evénement* est une action brusque, qui n'a pas de durée, comme un ordre envoyé par un *Acteur* extérieur au système, ou un *messages* diffusé par un objet du système, l'émetteur.

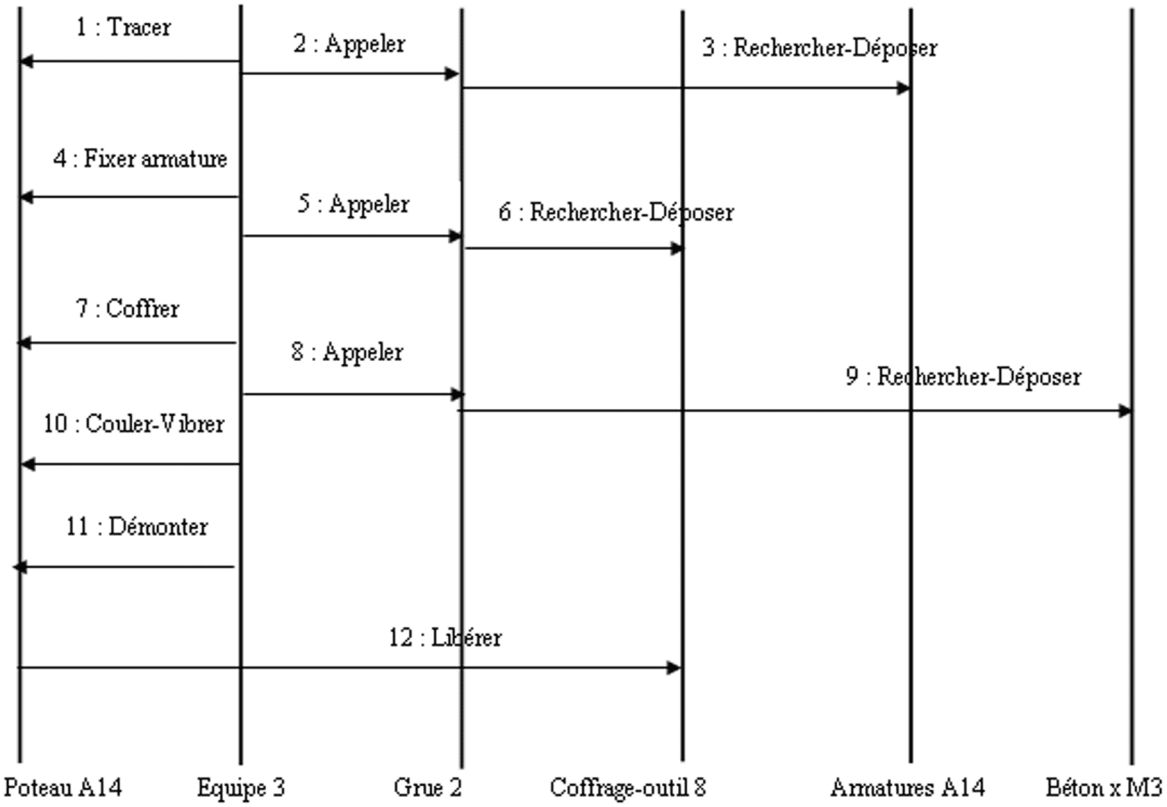
Cet ordre est obligatoirement donné à un objet du système, le récepteur. Le récepteur peut réagir ou non à cet ordre. Cela dépend du contexte. L'effet qui peut en résulter est soit le renvoi d'un autre message (la transmission de l'ordre à d'autres objets), soit l'exécution d'une action, ou soit un changement de l'état de l'objet qui a réagi à l'événement.

Il y a bien évidemment analogie avec les **langages à** *objets* : l'événement correspond à une **méthode**. Un objet d'une classe qui reçoit une demande d'exécution d'une méthode peut soit l'accepter si elle figure dans la liste des méthodes autorisées pour sa classe, soit la refuser dans le cas contraire. L'objet renvoie alors un message d'erreur à l'émetteur.

UML fait une distinction subtile entre un *messages*, qui est obligatoirement émis par un objet d'une classe, et un *Evénement*, qui englobe toutes les formes et tous les émetteurs, y compris les acteurs.

Chaque *Evénement* porte un nom, et il est représenté par un vecteur orienté, en trait mince.

Illustrons un *diagramme Scénario* par une fonction banale et apparemment simple sur un chantier : couler un poteau en BA. Le contexte est précisé : on utilise un coffrage-outil métallique, dont on doit gérer les permutations. La base de données du Bâtiment doit être capable d'accueillir l'information nécessaire pour des exploitations à toutes les étapes de la vie du bâtiment. Et la période du chantier est la plus critique et la plus riche. La fonction ne détaille les classes d'objets et les évènements que dans le but de d'identifier les postes analytiques d'un quantitatif estimatif. L'objet « Grue » intègre le grutier et son coût.



Au niveau de *granulométrie* requis, même si un événement semble prendre un certain temps, il peut être considéré comme nul : prendre, déposer, vider ... ou bien on considère que l'événement est l'ordre de commencer une tâche : tracer. Sous entendu, le chef d'équipe donne l'ordre aux ouvriers de l'équipe de commencer à tracer.

L'échelle du diagramme du scénario ne tient pas compte des durées. Exemple : vibrer le béton se fait juste après, et même pendant le coulage, tandis que démonter le coffrage ne se fait pas avant 21 jours.

Le débutant peut être hésitant sur l'émetteur et le récepteur d'un événement. Par exemple, pour exprimer le fait que la grue saisit une armature (sur le lieu de stockage) pour la déposer (sur une dalle) sur la localisation d'un poteau, ne doit-t-on pas relier l'objet grue à armature, puis grue à poteau ? (ce qui n'a pas été fait ici).

On verra plus loin, avec le concept plus précis de *Transition*, que l'on peut qualifier l'événement.

Par ailleurs, à la vue de ce diagramme, on pressent que le poteau en construction passe par plusieurs *Etat* ...

Par ailleurs, on imagine aussi que l'équipe 3 ne va pas se tourner les pouces pendant 21 jours avant de démonter le coffrage du poteau A14.

C. Le diagramme “d'Etats”

Un diagramme d'*Etat* est associé à une seule classe d'objets. Il est chargé de représenter tous les états possibles de ce type d'objets, et les *Transition* entre ces états.

Il porte donc le nom de la classe décrite. Deux nouvelles définitions sont nécessaires

pour le constituer :

Etat d'un objet.

Un *Etat* d'un objet d'une classe est défini par une stabilité de son comportement et de sa description pendant une durée déterminée. Cet espace de temps se délimite par deux *Evénement*.

Reprenons l'exemple du *diagramme Scénario* de construction du poteau, et isolons justement l'objet poteau.

Avant d'entrer dans le diagramme, le poteau était seulement en projet, repéré par la position A14 sur la trame d'un étage. Et encore avant, sa description dépendait de l'état d'avancement des études.

Le poteau passe donc successivement par les *Etat* suivants :

- Avant-projet (dimensions et position approximatives)
- Projet (dimensions, matériaux et position exactes)
- Projet-exécution (repéré dans une trame de localisation, il a fait l'objet d'un calcul de stabilité, et un plan d'armatures lui est associé)
- Tracé
- Armatures en place
- Coffrage en place
- Coulé et vibré
- En prise
- Décoffré
- Ragréé (le ragréage est l'opération de surfaçage d'un béton brut de décoffrage).
- Peint
- Livré
- En service

Et en prévision de la maintenance du bâtiment, on pourrait décomposer en service par :

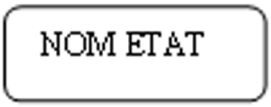
- En bon état (!)
- Abîmé
- Repeint

Cela ne fait pas moins de 14 états recensés pour un vulgaire poteau. Selon le niveau de granulométrie requis lors de l'analyse, on pourra regrouper des états, ou les décomposer.

Par exemple, à un niveau plus général, on pourra se contenter de 3 états :

- Poteau à l'état d'étude
- Poteau en construction
- Poteau en service

On note un *Etat* par un rectangle arrondi, portant le nom de l'état :

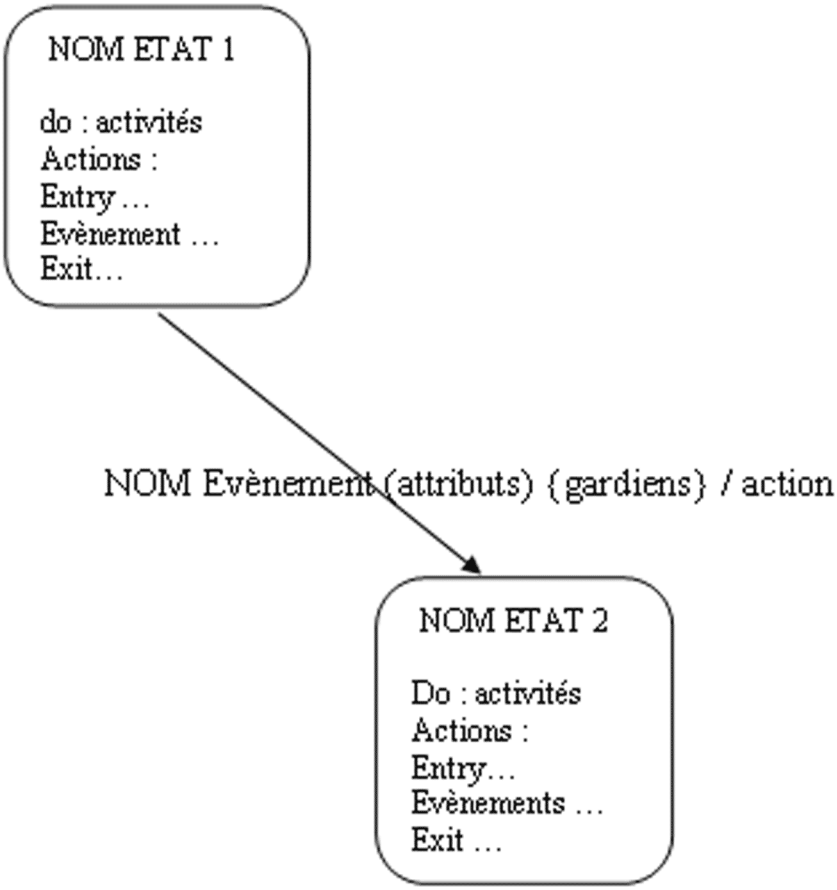


Transition entre deux états.
Une *Transition* est un changement d'*Etat* provoqué par un *Evénement*.

Il est donc naturel de le représenter par deux états, la flèche figurant l'événement défini plus haut, le nom de cet événement. Bien sûr, on pourra compléter ce nom par des paramètres qualifiant la transition : les *Attribut* et les *gardiens*. De plus, une *Transition* provoque également des *Opération*.

Un *Attribut* , représenté entre parenthèse après le nom de l'évènement, affine la description de la *Transition* (ce peut être une liste).

Les *gardiens*, représentés entre crochets, sont les conditions (booléennes) qui doivent être vérifiées pour que la *Transition* s'accomplisse (par exemple l'état d'autres objets de l'environnement, une variable calculée qui atteint la valeur limite).



Une (ou plusieurs) *Opération* peut être déclenchée par l'objet qui change d'état.

Dans le diagramme statique, les opérations, ou méthodes, sont notées après les attributs de la classe.

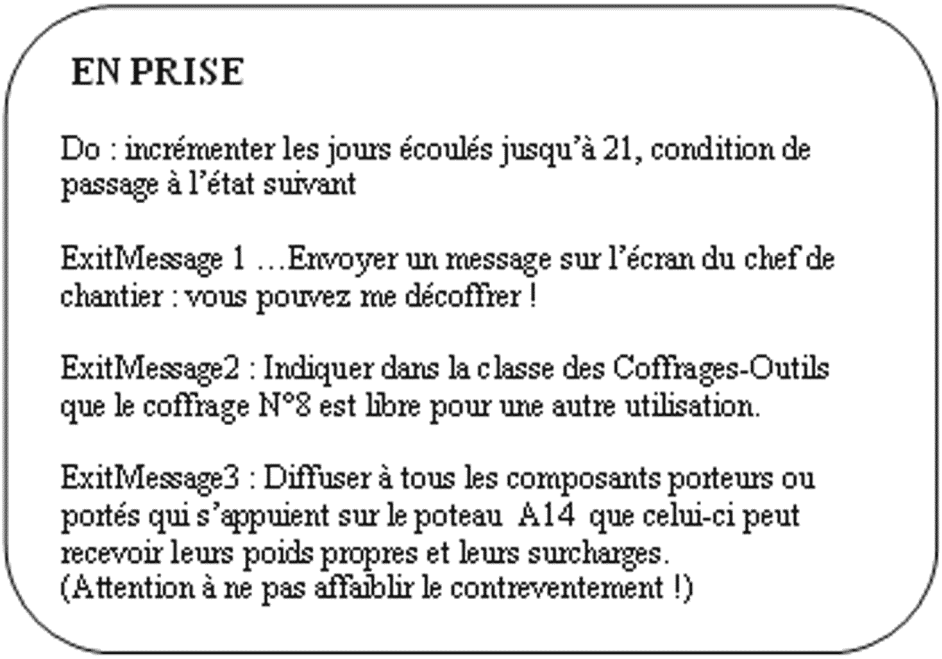
Dans le diagramme d'état, on distingue deux sous-types d'opérations qui seront inscrites à des endroits différents :

- les *activité* : c'est une opération répétitive, qui peut durer un certain temps. Une opération est donc par définition associée à un état et peut durer autant que lui. On l'inscrit dans le rectangle de l'état, précédé de la notation « do ».
- les *action* : c'est une opération ponctuelle, qui peut être réalisée à différents moments, associée à l'objet, ou à la transition :

- exécutée à l'entrée du changement d'état : on note le préfixe « entry »
- exécutée à la fin du changement d'état : préfixe « exit »
- exécutée à la demande d'un événement qui intervient n'importe quand : préfixe « Nom événement »
- exécutée pendant la transition elle même et par la transition : on note alors cette action dans les paramètres de l'événement, précédé du signe « / »

Dans l'exemple du poteau en construction, heureusement peu de paramètres sont à associer à chaque état. Sauf si l'objectif est d'imaginer le modèle d'un véritable logiciel d'assistance à la conduite du chantier.

Par exemple pour l'état en prise, on peut noter ainsi les conditions et opérations associées qui sont surtout importantes au moment de la fin de période de prise :



Le dernier message N°3 va provoquer une foule d'autres messages en cascade dans les objets d'autres classes de la base de données. Les poutres, dalles, qui étaient étayées vont changer de statut, ceux qui étaient encore à l'état de projet (le poteau d'au dessus par exemple) vont pouvoir être tracés ...

A ce niveau de *granulométrie*, on s'aperçoit par exemple qu'il faudra créer un état intermédiaire entre EN PRISE et DECOFFRE. En général, il n'y a pas beaucoup de place sur un chantier. Donc le coffrage N°8 pourra rester sur le poteau A14 jusqu'à réemploi, au delà des 21 jours. On évite d'encombrer l'aire de stockage, et une manipulation par la grue.

D'ou l'état supplémentaire : EN ATTENTE DE DECOFFRAGE.

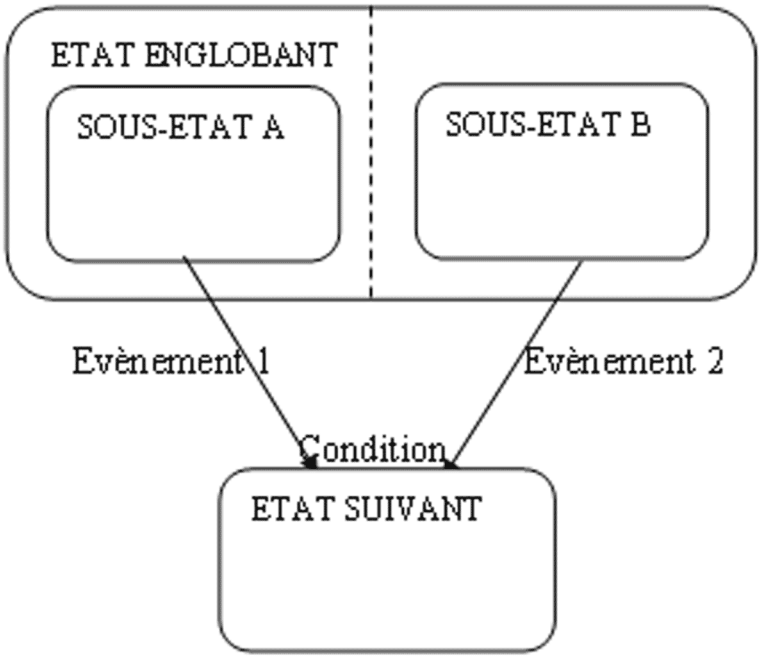
D. Sous-état, concurrence et synchronisation d'états.

Ce sont deux nouveaux concepts qui sont constamment vécus dans la vie courante !

Par définition deux états du même objet ne peuvent exister simultanément dans le temps.

Mais UML se tire de cette impasse en utilisant le concept de **sous**-*Etat* : Deux ou plusieurs **sous** *Etat* peuvent exister en parallèle.

On représente les **sous** *Etat* séparés par un trait en pointillé dans l'état englobant.



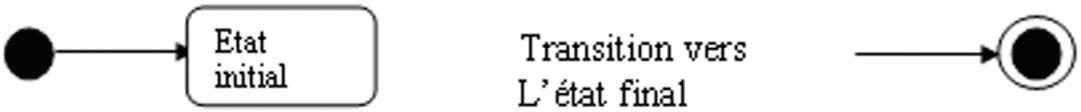
Un sous-état peut se terminer avant l'autre : c'est la *concurrence* : le premier évènement réalisé assurera la transition et décidera de l'état suivant. On note la condition par le « OU » exclusif.

Pour passer à l'état suivant, il faut que les deux évènements soient réalisés : c'est la *synchronisation*.

On note la condition par « ET ».

Etat initial, état final.

Dans un diagramme d'état, on distingue l'*Etat* **initial** qui est le point d'entrée du diagramme (donc un seul) par un rond noir, et l'*Etat* **final** (ou les états) par un rond noir entouré d'un cercle.



Vous en savez assez pour lire ou élaborer un diagramme d'états.

Exercice 1 : Diagramme d'états partiels du poteau.

Ne perdez pas de vue que vous devez exploiter ce diagramme d'état pour calculer automatiquement un quantitatif-estimatif analytique.

Vous pouvez imaginer un autre exercice : établir le diagramme d'état de l'occupation de l'équipe à la construction du poteau, en gardant à l'esprit qu'une équipe ne doit jamais rester sans rien faire (et pour un bon chef d'équipe, qu'aucun de ses membres ne soit inactif).

La dure réalité de la vie peut se modéliser !

Vous pouvez aussi imaginer, (mais alors c'est tout un programme !), que votre système d'information soit capable en cliquant sur une icône, de vous calculer instantanément les quantités exécutées, le respect ou les dépassements des délais, le personnel qu'il faut embaucher pour rattraper le temps perdu, éditer les situations, éditer la liste des approvisionnements pour les 30 jours suivants, ... et bien d'autres précieuses tâches.

E. Le diagramme de “Collaboration”

Son nom n'indique pas entre qui est envisagé la collaboration.

On peut comprendre ce concept de **collaboration** de deux façons :

- Collaboration entre plusieurs classes d'objets pour réaliser une fonction du système.
- Synthèse de certaines informations issues du modèle statique et des diagrammes dynamiques précédents qui collaborent pour offrir une vue partielle plus opérationnelle de la même fonction.

Un *diagramme de collaboration* assure la liaison entre les deux types de modèles.

Il contient une densité d'informations qui peut le rendre difficilement lisible.

On découpe les diagrammes en parties dont la granulométrie correspond à celle d'un scénario.

Pour une fonction donnée, et un degré de granulométrie donné, le *diagramme de collaboration* contient deux types d'information : le *contexte*, et les **interactions**.

Le contexte

Le *contexte* est tout simplement un extrait du modèle statique, un **package**, ou schéma qui ne réunit que les classes concernées, avec leurs **associations**, *Contraintes*, *Attribut*, *Opération*. Pour une meilleure lisibilité, on ne détaille pas le contenu d'une classe.

Si, comme nous l'avons déjà évoqué, de nouveaux objets de classes naissent à l'occasion de la synthèse des aspects dynamiques avec le modèle statique, on note le commentaire « new » associé à la classe créée.

Les interactions

Une fois le *contexte* mis en place, on recense dans les *diagramme Scénario* et d'*diagramme d'états* les *messages* qui sont nécessaires à l'exécution de la fonction, et on les ajoute au contexte. L'ensemble de ces messages forment ce qu'on appelle les **interactions**.

Une ambiguïté subsiste alors : la description statique de la classe ne renseigne pas sur les différents états possibles des objets de la classe. On ne peut tout dire dans un seul diagramme !

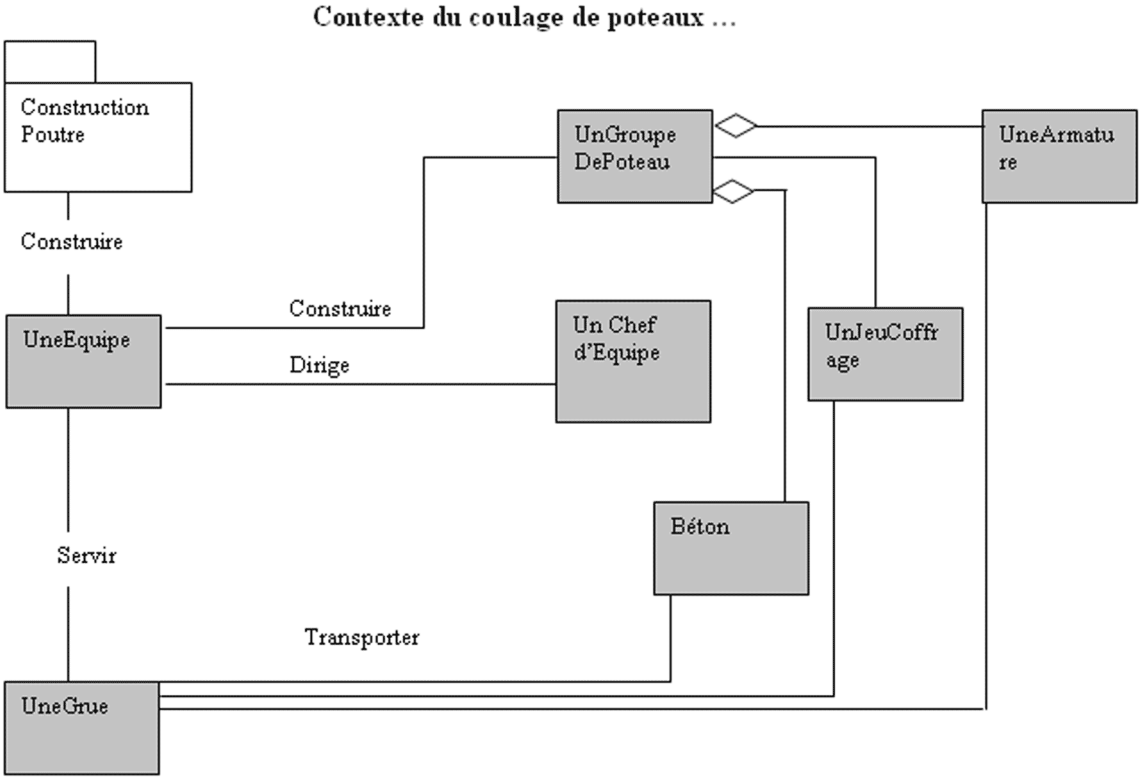
Cependant, on améliore la connaissance de l'écoulement du temps, par la description chronologique des messages reportés dans le diagramme. Il suffit de les numéroter.

Mais pas seulement avec un simple numéro d'ordre. Il convient de noter les *messages* en *concurrence*, ce qui permettra de les dérouler dans une arborescence séparée qui permettra aux futurs logiciels d'exploitation une exploration algorithmique d'un arbre.

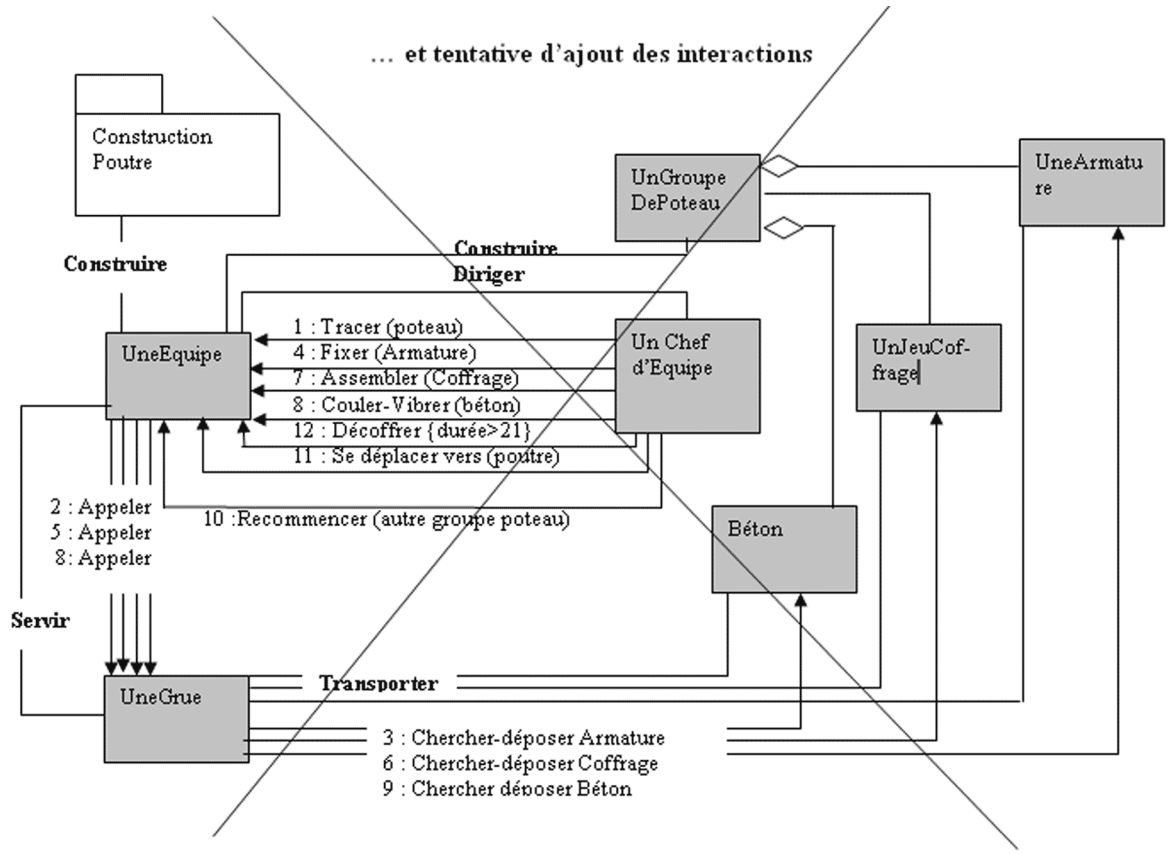
Pour bien comprendre, construisons un *diagramme de collaboration* en trois étapes :

Tout d'abord, par exemple, isolons quelques entités du scénario de construction d'un poteau.

Dans le diagramme qui suit, nous avons séparé l'équipe et son chef, qui émet les messages, ce qui est plus conforme à la réalité semble-t-il, que dans le scénario. Une analyse d'occupation des tâches a déterminé qu'une équipe devait s'occuper de 4 poteaux simultanément. Après le coulage, l'équipe a le choix entre passer à un nouveau groupe, ou bien passer à la construction de poutres, dont la présence est décrite dans le schéma par un package. Aux classes Armatures, jeu de coffrage, béton sont en fait associés des hommes qui obéissent au grutier.



Contexte du coulage de poteaux ...



... et tentative d'ajout des interactions

F. Ne pas se tromper de cible : faire porter l'information par les acteurs ou par les objets ?

Examinons le précédent *diagramme de collaboration*.

Il reflète la réalité de ce qui se passe sur le chantier : Le chef orchestre, c'est le chef d'équipe qui donne ses ordres à chacun de ses hommes et par délégation au grutier.

Ce diagramme serait pertinent si l'objectif était d'étudier uniquement l'enchaînement des tâches d'occupation d'une équipe, son emploi du temps détaillé. Dans ce cas, il est vrai que c'est l'équipe et son chef en tant que classes d'objets qui doivent porter l'information.

L'équipe change d'état à chaque changement d'activité. Le poteau, non porteur d'informations, n'est pas impliqué dans les *Evénement* !

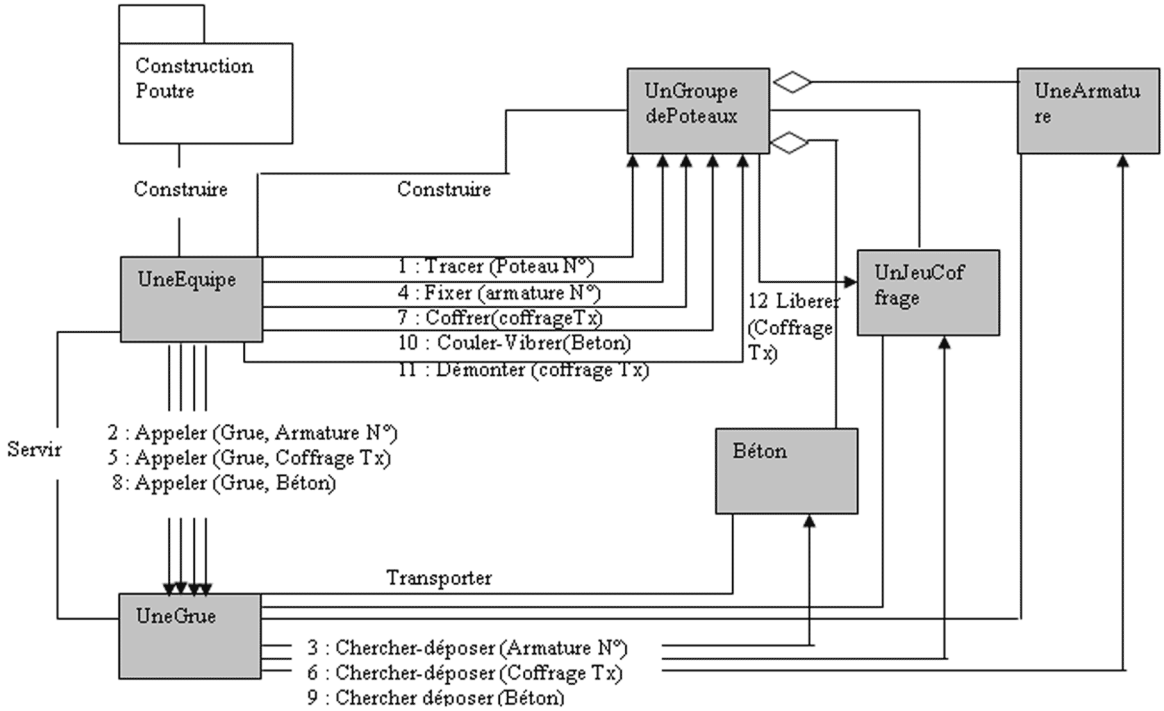
Mais notre objectif n'est pas celui-là. Revenons au **scénario** « couler un poteau en béton ».

Ce qui nous intéresse, c'est de pouvoir gérer les objets du chantier pour la rotation des coffrages, mettre à jour les travaux à exécutés, vérifier les plannings d'avancement, calculer le quantitatif, les situations instantanées, les stocks, les commandes ...

Ce sont donc les classes de ces objets qui doivent porter l'information, déclencher des *action*, même si en réalité ils sont inertes. La modélisation du Bâtiment ne décrit plus la réalité ergonomique ou humaine du chantier, mais une réalité comptable !

C'est une faute fréquente chez le débutant de se tromper de cible, et de confondre l'acteur et ses objets.

Le diagramme de collaboration fidèle à notre scénario est donc le suivant :



Les *Evénement* changent l'état du poteau, lequel libère en dernier le coffrage (Type x) en le marquant disponible dans sa classe.

Dans un diagramme où les ressources en main d'œuvre ne sont pas gérées, on pourrait intégrer la grue à l'équipe, et même supprimer l'équipe du diagramme.

Ce serait alors à la classe des poteaux de gérer par des messages les classes Armatures et Béton. C'est l'objet de l'**exercice 2** à l'extérieur de cette unité, **dans les exercices en annexe**.

Nota : Nous avons noté les messages par une numérotation continue de 1 à 12.

En utilisant une numérotation en arbre, on peut traduire un diagramme de collaboration en une arborescence de messages. Cela n'a d'intérêt que pour gérer les évènements en concurrence.

C'est une représentation très utile pour les informaticiens.

Par exemple, après avoir coulé un groupe de 4 poteaux, l'équipe peut selon le contexte

- recommencer un autre groupe de poteaux
- décoffrer un groupe de poteaux coulés il y a plus de 21 jours,
- se déplacer pour couler des poutres ...

ChapitreIV. Pour approfondir : les diagrammes conceptuels des IFC exprimés en EXPRESS-G.

A. Le formalisme de spécification formelle Express-G.

La totalité des informations du modèle IFC figure dans des schémas conceptuels reliés les uns aux autres (24 dans la révision 2). Ils sont difficiles à lire parce que justement toutes les entités y figurent, exprimées avec le détail requis pour les informaticiens.

En tant qu'utilisateur des IFC, vous n'êtes pas obligés de savoir lire ces schémas.
Mais si votre objectif professionnel est de devenir expert des outils informatiques en architecture, ingénierie ou construction, ce qui vous oblige à dialoguer avec par exemple des développeurs d'interfaces IFC, cette compétence est indispensable, nous l'avons souligné tout au long de ce cours.

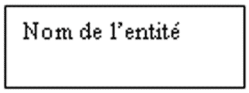
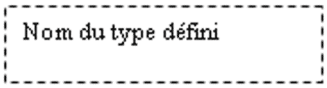
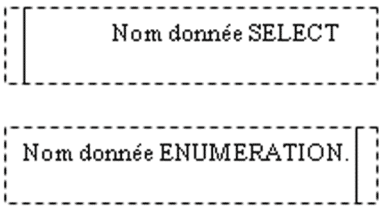
Les auteurs du modèle (l'IAI) ont privilégié le langage de description de base de données normalisé ISO *EXPRESS*. Il était donc naturel qu'ils choisissent un formalisme de spécification graphique en relation avec ce langage. C'est *EXPRESS-G*, qui autorise l'usage d'outils logiciels de traduction de diagrammes directement en instructions de langage.

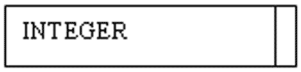
Ce qui ne fait pas votre affaire, car ce choix vous oblige, cher lecteur, à apprendre une méthode de spécification de plus !

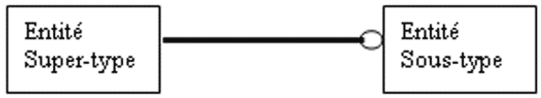
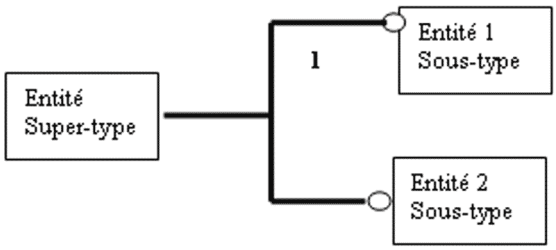
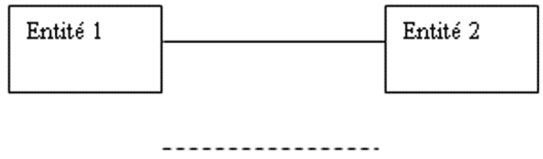
Mais ce sera continuellement le cas durant toute votre vie professionnelle ! Autant vous habituer tout de suite à circuler d'une méthode à l'autre. Ce qui n'est pas bien difficile, car les concepts fondamentaux sont toujours les mêmes. Et puis, possédant déjà les méthodes *NIAM* et *UML*, ce sera pour vous un jeu d'enfants !

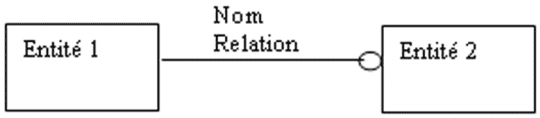
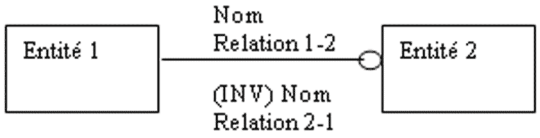
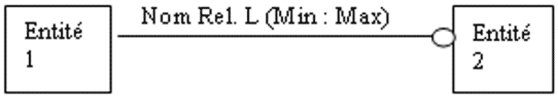
Nous donnons ci-après la syntaxe graphique d'*EXPRESS-G*, avec quelques commentaires. Pour plus de détails, vous pouvez consulter les documents normalisés ISO de STEP : l'annexe D du volume 11.
Vous trouverez à l'extérieur de cette unité un fichier PDF imprimable "SyntaxeGraphiqueExpressG.pdf" pour vous permettre de mieux visualiser ces informations

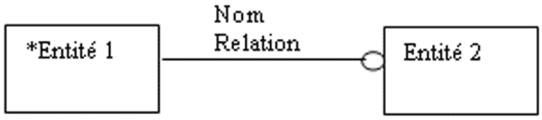
qu'à l'intérieur de cette page web. Le contenu de ce fichier PDF peut être agrandi à l'écran.

Concept et syntaxe Express-G	Commentaires
Entité (entity-name) 	Classe d'objets dans UML. Concept dans NIAM L'entité peut être un attribut
Type défini (new-defined-data-type) 	Type d'entité très utilisé dans les IFC Correspond au type défini de l'arborescence en annexe, qui est une spécialisation de la classe décrite, et dont le nom est normalisé par les IFC ou les utilisateurs.
Type de données construit : SELECT et ENUMERATION 	Donne des indications sur la manière dont on construit ces données par sélection ou énumération à partir d'une classe. Pour les informaticiens

Concept et syntaxe Express-G	Commentaires
Type de données simple Par exemple : 	Indication sur le type de données pour les informaticiens (type entier, réel, chaîne de caractère)
Relations On distingue dans l'ordre d'importance : Les relations d'héritage Les relations entre entité Les relations de type attribut	On retrouve les mêmes concepts dans NIAM et UML, avec des nuances

Concept et syntaxe Express-G	Commentaires
Héritage simple 	le sens de la relation est donné par Le petit cercle placé contre Le rectangle de l'entité sous-type Le trait de la relation est épais
Héritage avec contrainte d'exclusion 	Pour exprimer que l'on doive choisir un seul des deux descendants à la fois, (contraintes d'exclusion dans NIAM) On marque le chiffre un sur l'arborescence Comme dans NIAM et UML, l'héritage Multiple n'est pas formalisé.
Relation simple 	Dans le cas général, c'est un trait Mince entre les deux entités. Sans autre indication, il n'y a pas de de sens privilégié dans la relation Exemple : relation entre un objet et un **Attribut**. Pas de nom à cette relation. Trait en pointillé si **attribut optionnel**.

Concept et syntaxe Express-G	Commentaires
Relation à sens prioritaire 	C'est la majorité des cas. On précise toujours le nom de la relation dans le sens prioritaire, qui s'identifie ainsi à l'un des deux Rôle dans NIAM
Relation qualifiée dans les 2 sens 	On précise en plus le nom de la relation dans le sens inverse au sens prioritaire. Ainsi l'information est équivalente aux deux Rôle de NIAM ou UML. Mettre le nom sous le trait, précédé du sigle (INV)
Contraintes de cardinalités 	Par défaut, au moins 1, donc plusieurs. C'est différent de NIAM (0 autorisé). En général, après le nom de la relation, on place entre parenthèse les cardinalités minimales et maximales, précédé d'une lettre dont le sens est réservé aux informaticiens : Tableau, Liste … ? signifie que le Max n'est pas défini.

Concept et syntaxe Express-G	Commentaires
Contrainte autres que cardinalités 	Quand une entité fait l'objet d'une Règle dans une relation, on inscrit un astérisque précède le nom de de l'entité objet ou attribut

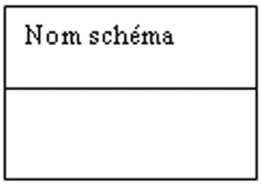
D'autres nuances peuvent être exprimées pour le type des attributs. Nous n'entrerons pas dans le détail d'Express-G.

B. La circulation entre les pages d'un schéma Express-G.

Vous trouverez constamment dans les diagrammes Express-G cinq symboles graphiques supplémentaires uniquement destinés à permettre le découpage d'un grand schéma en plusieurs parties pour qu'elles puissent entrer dans une page papier, et être lisibles.

Dans le cas des IFC, il existe de nombreux schémas, qui portent un nom.

Schéma

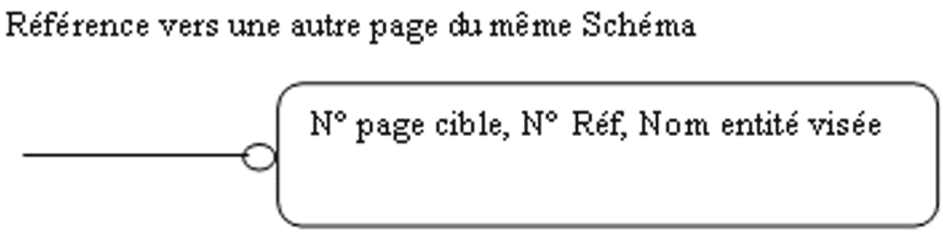


Chaque schéma doit
porter un nom pour être référencé
dans les autres schémas

Schéma

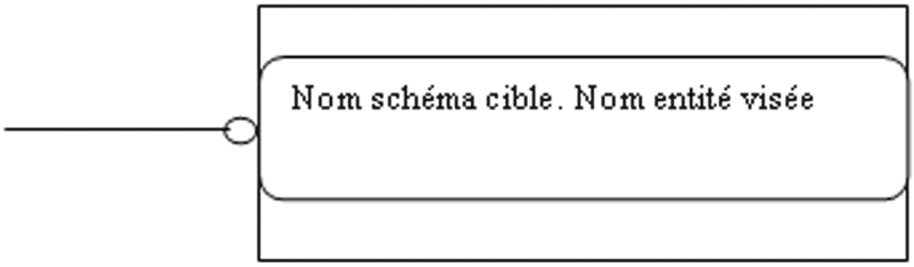
Un schéma peut être découpé en plusieurs pages. Dans une page, on peut indiquer que l'on relie une entité (dite source) vers une autre (dite cible), et dans ce sens prioritaire,

- située sur la même page (à éviter, on peut toujours se débrouiller pour tracer un trait entre deux entités de la même page !),
- située sur une autre page du même schéma,
- située dans un autre schéma. On « utilise une définition » du schéma cible.



Référence vers une autre page du même Schéma

Utilisation d'une entité décrite dans un autre schéma

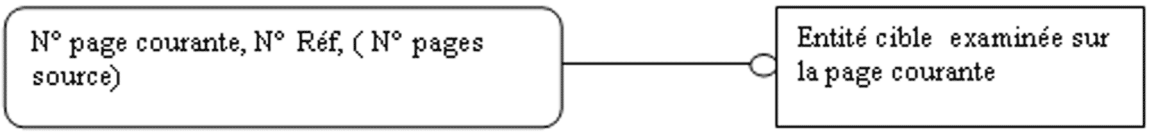


Inversement, une entité de la page que l'on examine (la page courante) peut être reliée à une autre entité d'une autre page, ou d'un autre schéma, mais dans le sens inverse de la relation.

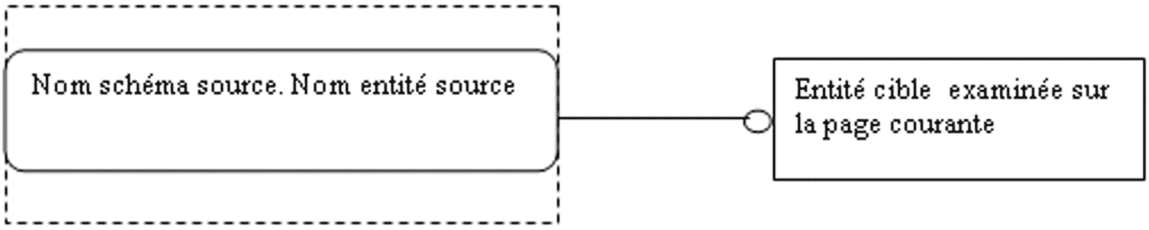
On dira que l'entité examinée (cible) reçoit une référence, une définition en provenance d'une entité source.

Référence en provenance d'une autre page

Sur la page courante examinée, on trouve une ou plusieurs références toutes précédées du numéro de la page courante :

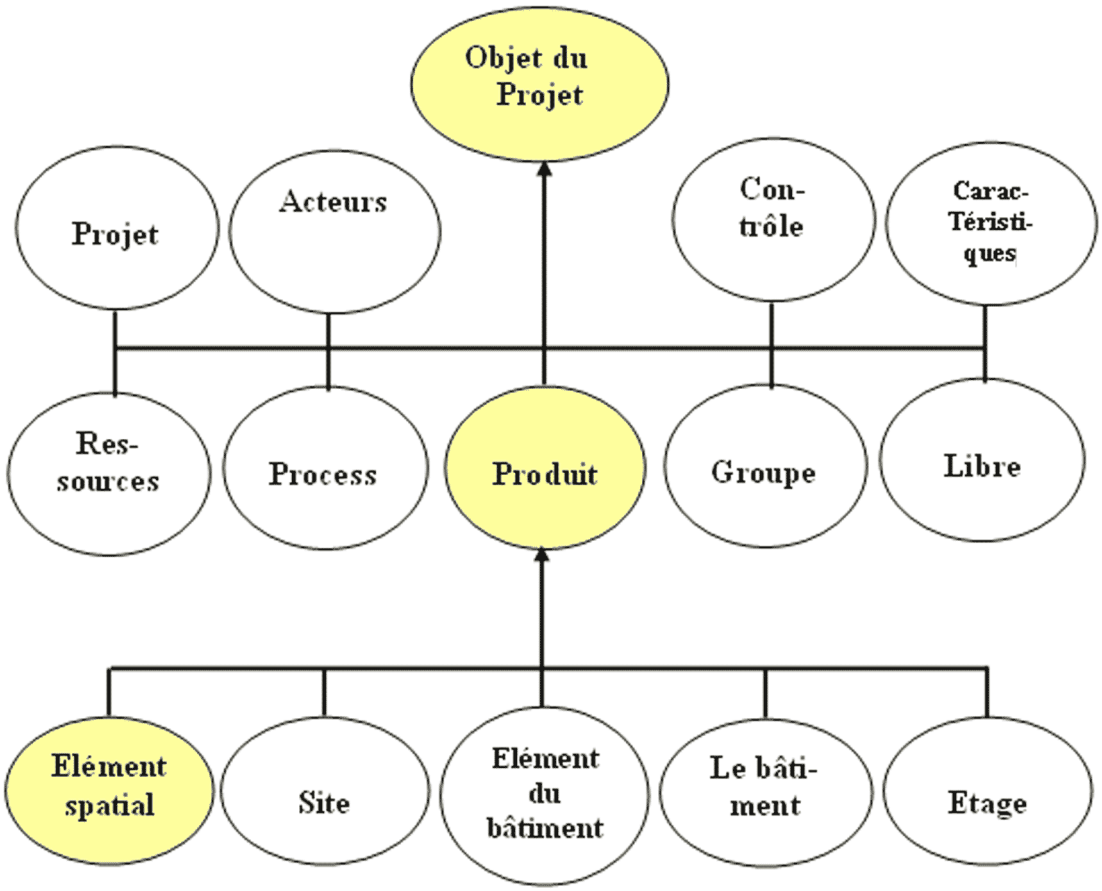


Référence en provenance d'un autre schéma



C. Début du parcours d'un schéma Express-G des IFC : l'arbre général

Revenons à l'arborescence générale d'héritage des IFC décrite dans l'unité 3, et intéressons-nous cette fois à la classe « *Elément spatial [ref. 2]* ».



Cette classe « *Elément spatial [ref. 2]* » appartient au niveau de l'architecture des IFC dénommé *IfcProductExtension [ref. 5]* représenté par un schéma IFC qui décrit les objets issus de l'objet *Produit [ref. 14]* .

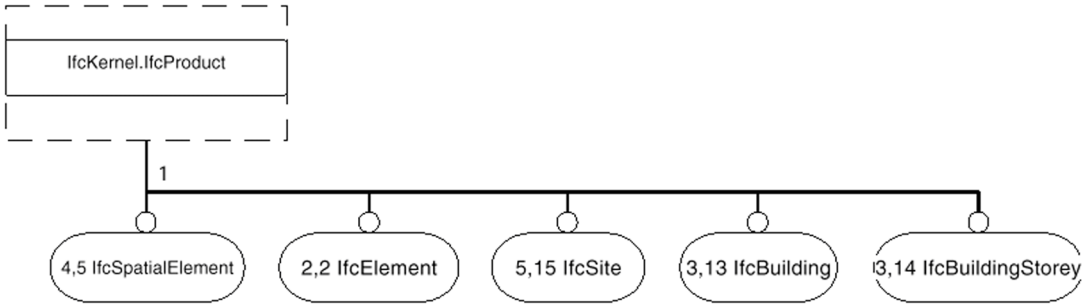
Ce schéma contient 7 pages de diagrammes, reliées entre-elles, et bien évidemment souvent reliées à bien d'autres schémas.

Ensemble, nous nous proposons d'examiner uniquement toutes les entités concernées par l'Elément spatial, pour en connaître les liaisons entretenues avec les entités de son environnement, informations qui n'apparaissent pas dans les arbres d'héritage du modèle.

Vous ne pourrez donc pas voir ces informations dans l'annexe arborescence des principales entités des IFC, mais uniquement dans les schémas conceptuels.

Pour ce faire, circulons dans le schéma *IfcProductExtension [ref. 5]* **depuis sa racine.**

Dans la Page 1 du Schéma *IfcProductExtension [ref. 5]*



Un petit diagramme sert uniquement à lister l'héritage de l'objet *Produit [ref. 14]* , dont l'objet *IfcSpatialElement [ref. 10]* , qui est décrit dans la page 4, avec la référence 5.

D. ... Puis examinons la page 4 du Schéma IfcProductExtension

Affichons cette page 4.
On remarque qu'un *Elément spatial [ref. 2]* se spécialise en *Local [ref. 11]* (*IfcSpace [ref. 8]*) ou en *Nu de local [ref. 13]* (*IfcSpaceBoundary [ref. 9]*) (Relation d'*Héritage*).

Le *Local [ref. 11]* comprend un certain nombre d'attributs optionnels (périmètre, hauteur brute et nette, volume, altitude, référence, nom, surface) décrits dans d'autres schémas (Celui des *ressources [ref. 15]*). L'attribut obligatoire (Intérieur ou extérieur) est un **type de données construit**. Le local entretient deux types de relations :

1. une relation que l'on peut qualifier de voisinage avec ses*Nu de local [ref. 13]* (*IfcSpaceBoundary [ref. 9]*). Les cardinalités de la relation indiquent qu'un local peut avoir plusieurs *Nu de local [ref. 13]* , et au moins 1 (Un dans le cas sans doute où le local est une sphère, et 6 si c'est une pièce à 4 murs).
2. Mais c'est la relation inverse qui est la plus intéressante : Un *Nu de local [ref. 13]* ne peut avoir qu'au maximum 2 *Local [ref. 11]* voisins, donc **un seul de chaque coté**.

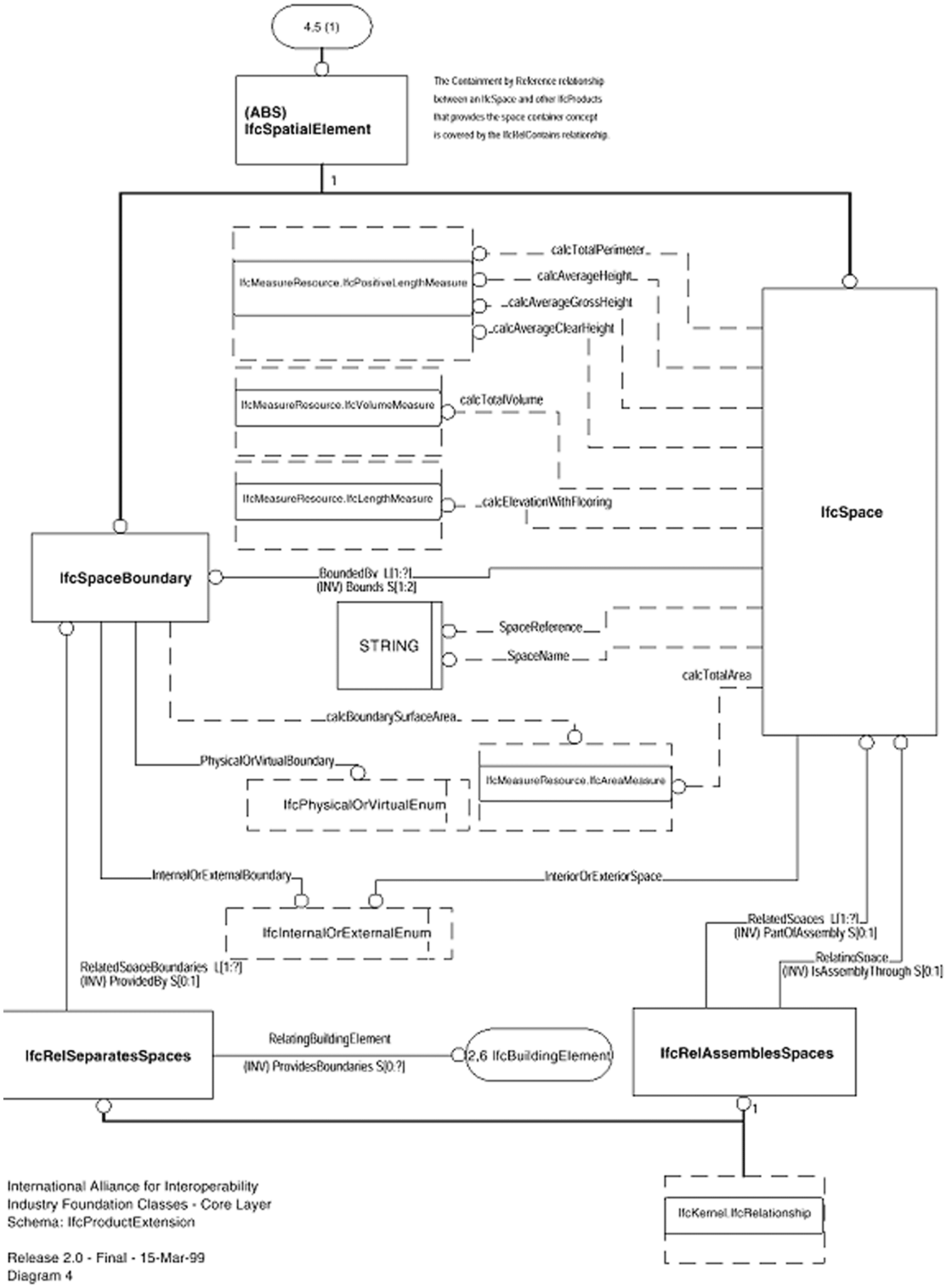
   Cette toute petite indication noyée dans un schéma est lourde de conséquence. .

Elle signifie que :

A retenir

Le modèle IFC suppose que les frontières délimitant les espaces sont munies de la propriété d*'homogénéité*. **Cette propriété s'étend-elle aux murs et planchers ?**

Vous trouverez à l'extérieur de cette unité, dans les fichiers complémentaires aux unités de cours version web, un fichier PDF imprimable "Page4_Schema_IfcProductExtension.pdf"et dont le contenu peut être agrandi. Pour vous permettre de mieux visualiser cette page 4 que dans cette page web.



Page 4 du Schéma IfcProductExtension

Nous avons examiné dans l'unité "Comprendre et échanger les vues métiers du Bâtiment" du cours cette propriété d'*'homogénéité* qui est fondamentale pour la faisabilité des échanges.

Il est évident que si l'on n'étend pas cette propriété IFC au moins aux *murs [ref. 12]* qui se confondent aux *Nu de local [ref. 13]* , ou qui selon les interprétations, supportent les Nus de local, on ne retire aucun intérêt de cette propriété.

Cette propriété est difficile à mettre en oeuvre par les éditeurs de logiciels techniques. De plus, ils ne sont pas habitués à ce concept qui oblige à de lourdes contraintes dans l'organisation des données de leur *modèle conceptuel* **interne**. Il faudra donc que vous, utilisateur averti, soyez très vigilant au respect de cette contrainte dans un cahier des charges, et dans les fonctionnalités d'un logiciel que vous achetez.

Les deux autres relations que l'objet Local entretient avec son voisinage sont formalisées par un objet **relation** (*IfcRelAssemblesSpaces [ref. 6]*) (n'oubliez pas que dans les IFC, une relation peut être considérée comme un objet, ce qui permet de lui associer à son tour des propriétés et des relations).

La première relation sur l'objet relation concerne une agrégation : un Local appartient à un ensemble plus vaste.

La deuxième relation sur l'objet relation est une relation de qualification de voisinage, par exemple de passage : il existe une possibilité de passage avec une liste de *Local [ref. 11]* voisins.

Ces propriétés relationnelles, les listes, sont décrites par l'objet *IfcRelationship [ref. 7]* qui figure dans le schéma racine « *IfcKernel [ref. 4]* ». Par cet intermédiaire, tous les objets IFC peuvent en théorie être reliés à tous les autres, selon les besoins de l'évolution du modèle.

A titre d'exercice informel, le lecteur détaillera lui même les relations et attributs de l'objet *Nu de local [ref. 13]* (*IfcSpaceBoundary [ref. 9]*).

Remarquez que la description de l'objet *IfcRelAssemblesSpaces [ref. 6]* se poursuit par l'entité Element de construction (*IfcBuildingElement [ref. 3]*) qui se trouve sur la page 2, avec la référence 6.

E. ... Puis la page 2 du Schéma IfcProductExtension

Affichons cette nouvelle page.

Dans le modèle IFC, le nom de la relation qui relie le *Nu de local [ref. 13]* (page 4) à l'**Elément de construction** (*IfcBuildingElement [ref. 3]*) (page2) n'est pas très explicite : s'agit-t-il d'une mise en coïncidence exacte des deux objets, ou bien seulement l'un s'appuie sur l'autre ?

La relation inverse nous renseigne plus : (INV)ProvidesBoundarie S(0 : ?)

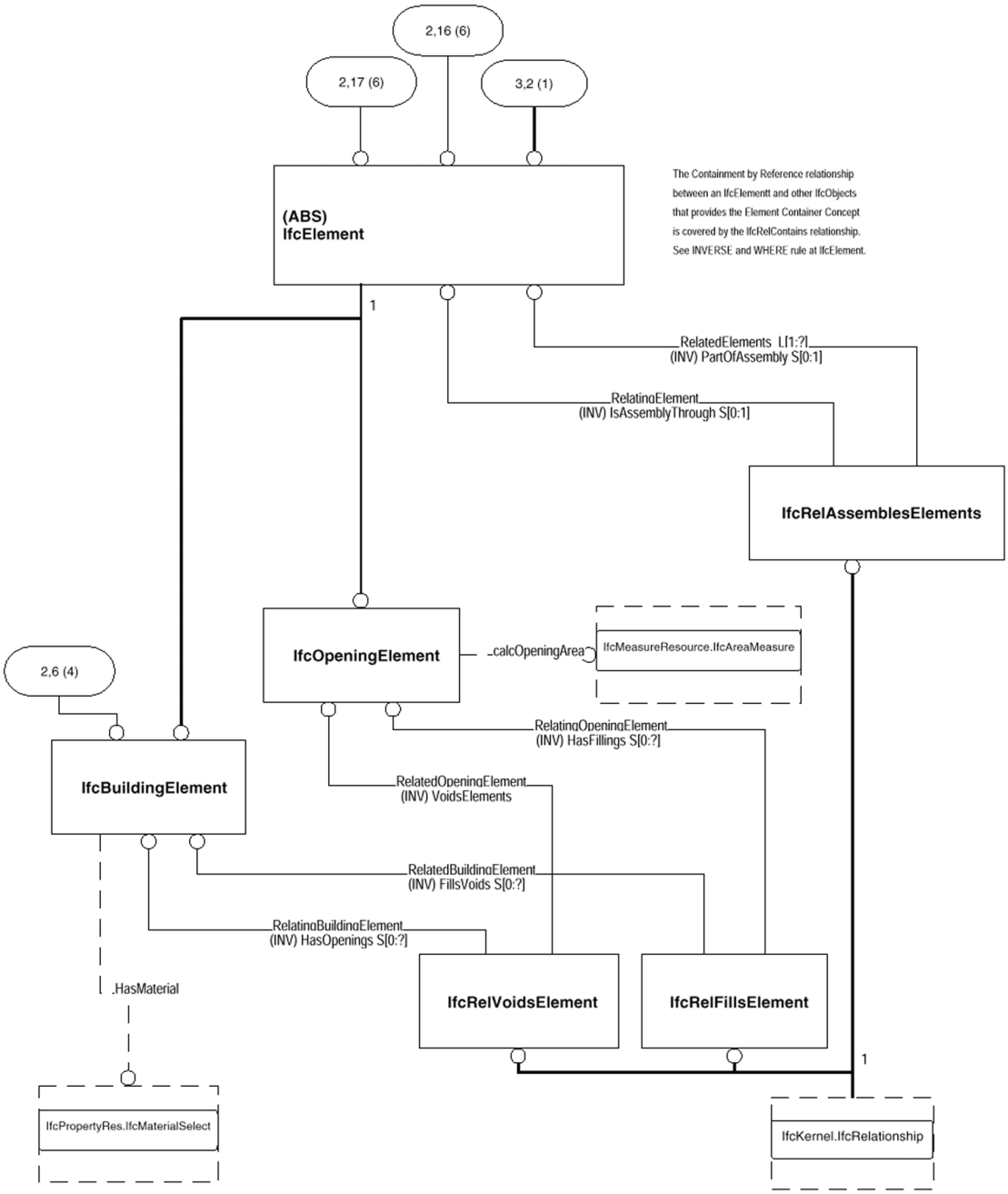
Le terme « provides » (fournit) et les cardinalités semblent indiquer que les deux objets existent avec des dimensions géométriques qui ne se recouvrent pas. Le Nu de local peut exister seul sans le support matériel d'un mur ou d'un plafond (0), et le point d'interrogation implique qu'il puisse exister plusieurs Murs sur un Nu de Local.

Donc le modèle IFC a décidé :

A retenir

Le Modèle IFC n'oblige pas à un re-découpage homogène des parois pour une exacte coïncidence avec les *Nu de local [ref. 13]* .
La propriété d'*'homogénéité* **des parois est une option à l'initiative de l'utilisateur.**

Cette propriété sera prise en compte « implicitement » par le modèle si la géométrie des parois et des *Nu de local [ref. 13]* **produite par un logiciel de CAO ou technique en tient compte. On peut seulement regretter cette liberté qui posera des problèmes dans les échanges de données (voir unité "Comprendre et échanger les vues métiers du Bâtiment" du cours).**



International Alliance for Interoperability
Industry Foundation Classes - Core Layer
Schema: IfcProductExtension

Release 2.0 - Final - 15-Mar-99
Diagram 2

Page 2 du Schéma IfcProductExtension

Vous trouverez à l'extérieur de cette unité, dans les fichiers complémentaires aux unités de cours version web, un fichier PDF imprimable "Page2_Schema_IfcProductExtension.pdf" dont le contenu peut être agrandi. Pour vous permettre de mieux visualiser cette page 2 que dans cette page web.

Le lecteur remarquera qu'un **Elément de construction** est en relation avec un *Nu de local [ref. 13]* .

Un *Nu de local [ref. 13]* est en relation avec un *Local [ref. 11]* .

Par ailleurs, dans la page 2, un **Elément de construction** est en relation avec les équipements d'ouverture qu'il contient.

Un **Elément de construction** est une sorte d' "Elément du bâtiment" qui est en relation avec potentiellement tous les objets que l'on désire, à travers l'objet relation *IfcRelationship [ref. 7]* situé dans le schéma racine *IfcKernel [ref. 4]* .

Remarquer que l'objet « *Element [ref. 1]* **» reçoit des descriptions en provenance de la page 6, qui décrit justement des relations dites « de calage » et de « coïncidence géométrique » (ce qui permet de décrire la topologie des axes filaires : voir unité de cours 2).**

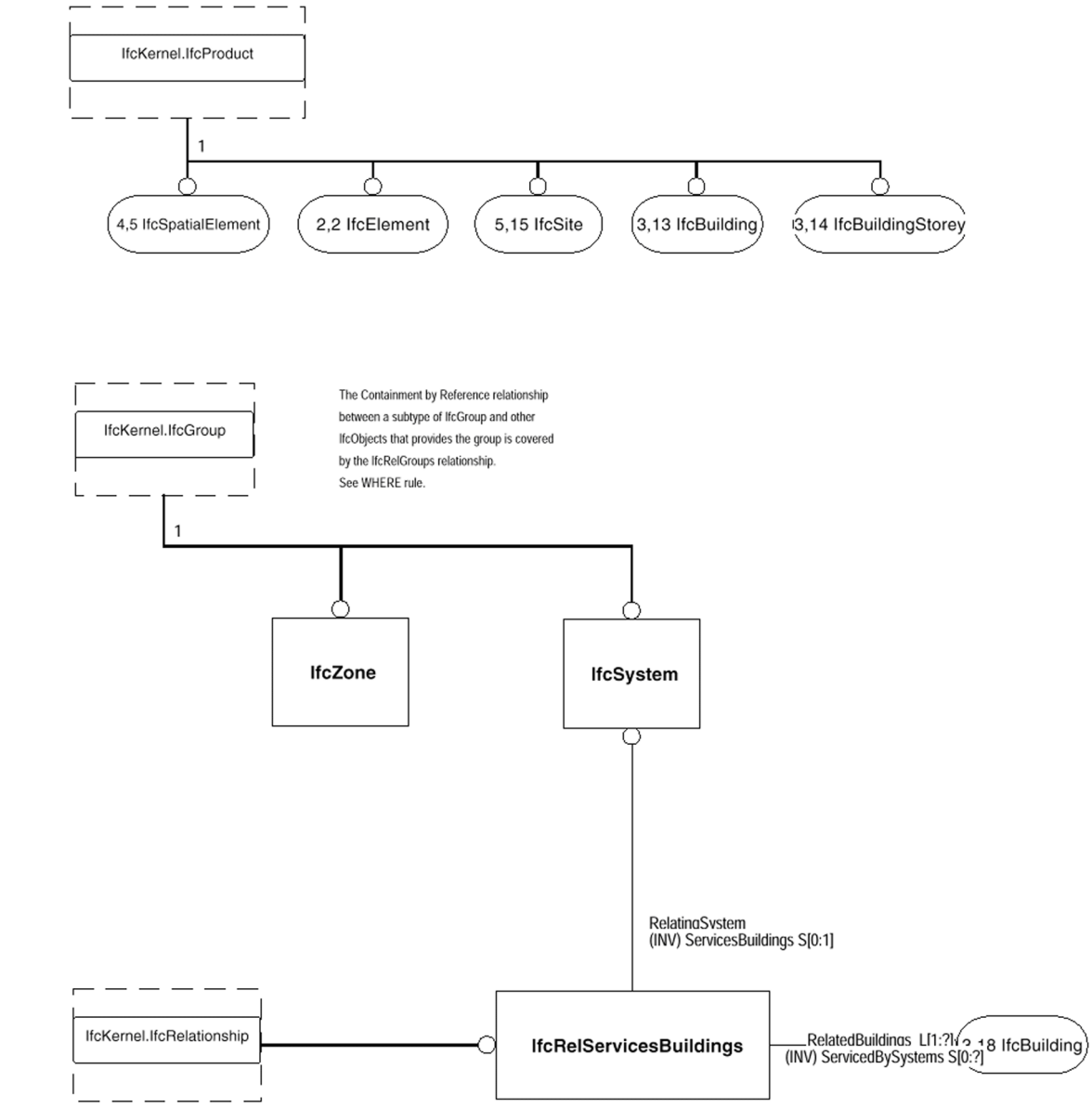
Par recherche transitive, on peut formuler dans la base de données d'échange IFC toute requête sur le voisinage d'un objet. L'examen d'un des 24 schémas du modèle en apporte déjà ici un début de preuve.

Nous proposons au lecteur un dernier exercice en dehors de cette unité, en annexe,(**Exercice "Lister les informations portées par une classe dans les pages du schéma IfcProductExtension"**) qui est le prolongement de l'exercice "Diagramme d'Etats partiels d'un poteau" : par un parcours de schémas, lister toutes les informations relatives à un thème.

F. ... Puis les autres pages du Schéma IfcProductExtension

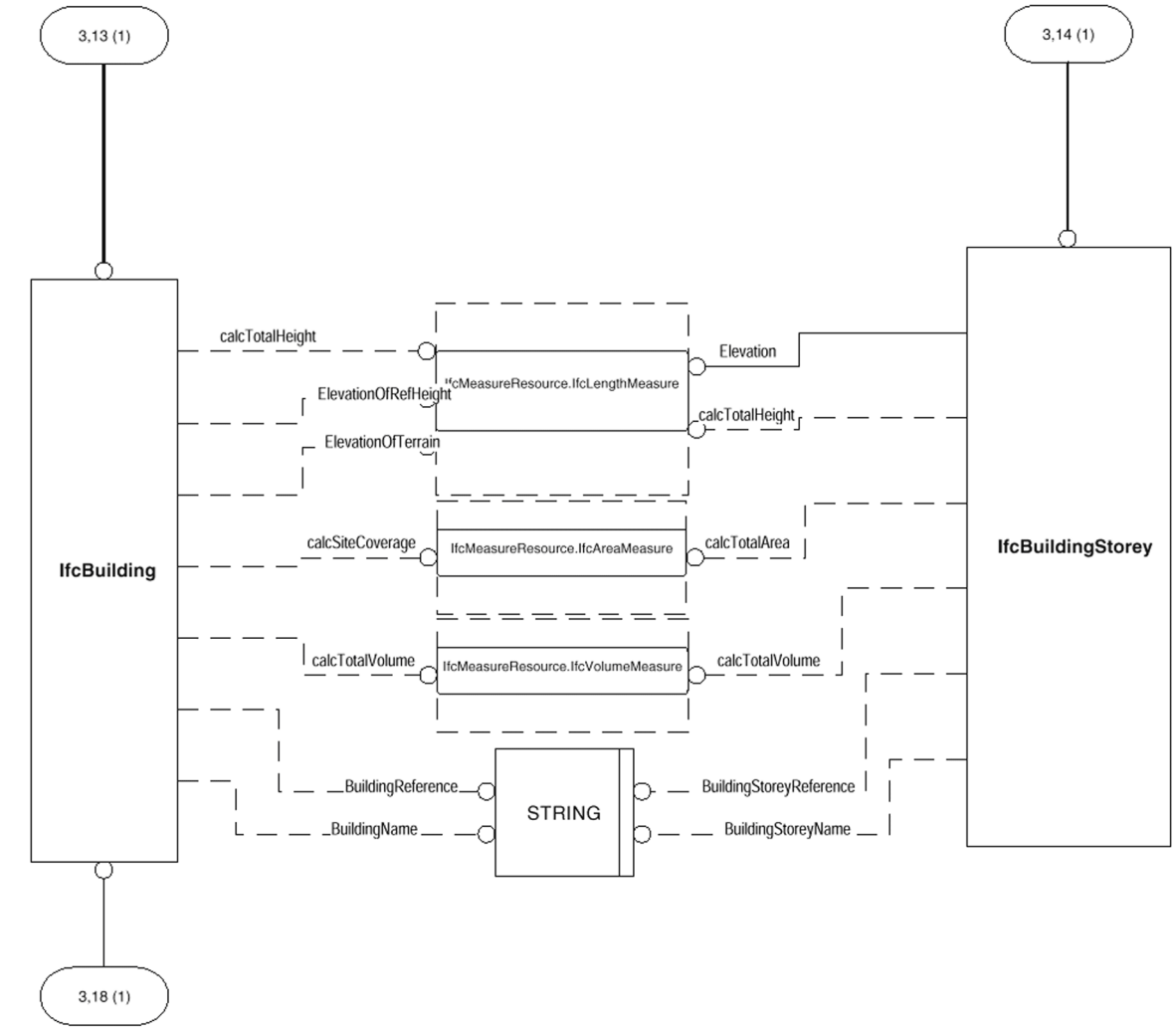
Vous trouverez à l'extérieur de cette unité, dans les fichiers PDF complémentaires aux unités de cours version web, un fichier PDF imprimable "Schema_IfcProductExtension.pdf" dont le contenu peut être agrandi. Pour vous permettre de mieux visualiser ces pages que dans cette page web.

Page 1 du Schéma IfcProductExtension



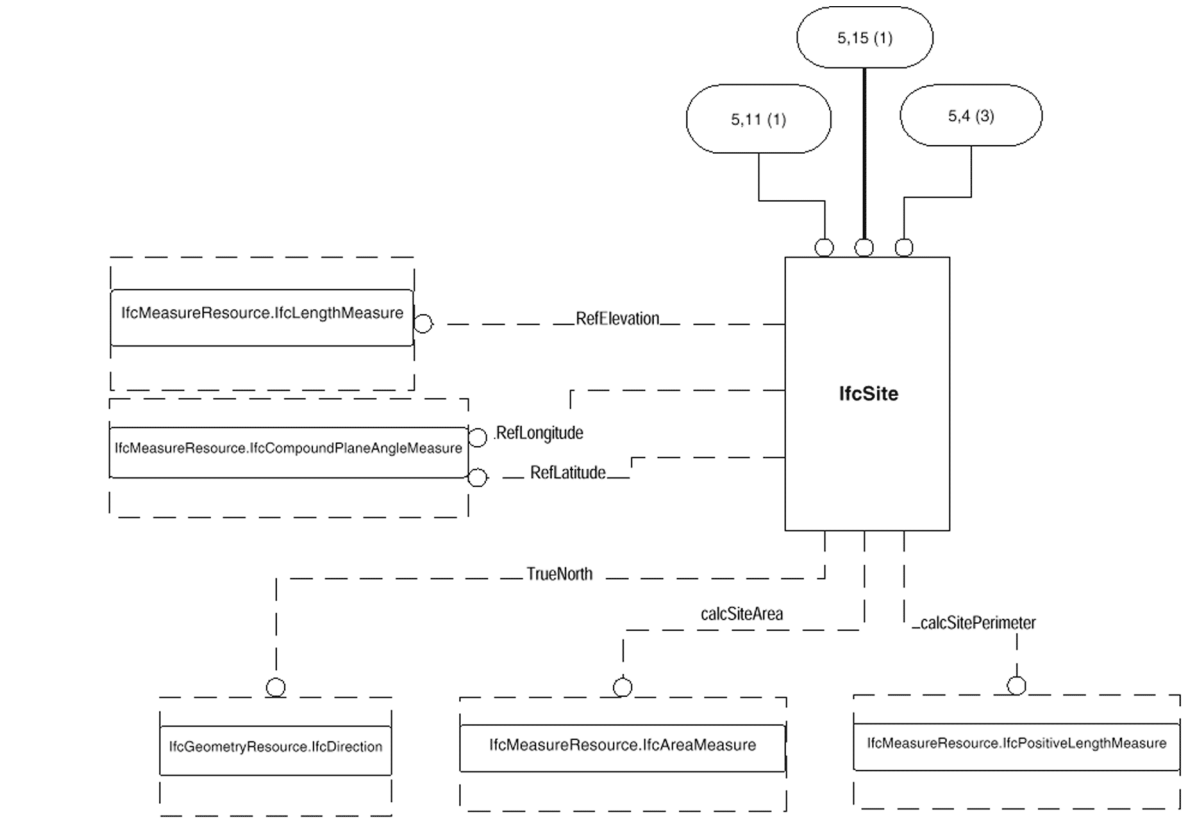
Page 1 du Schéma IfcProductExtension

Page 3 du Schéma IfcProductExtension



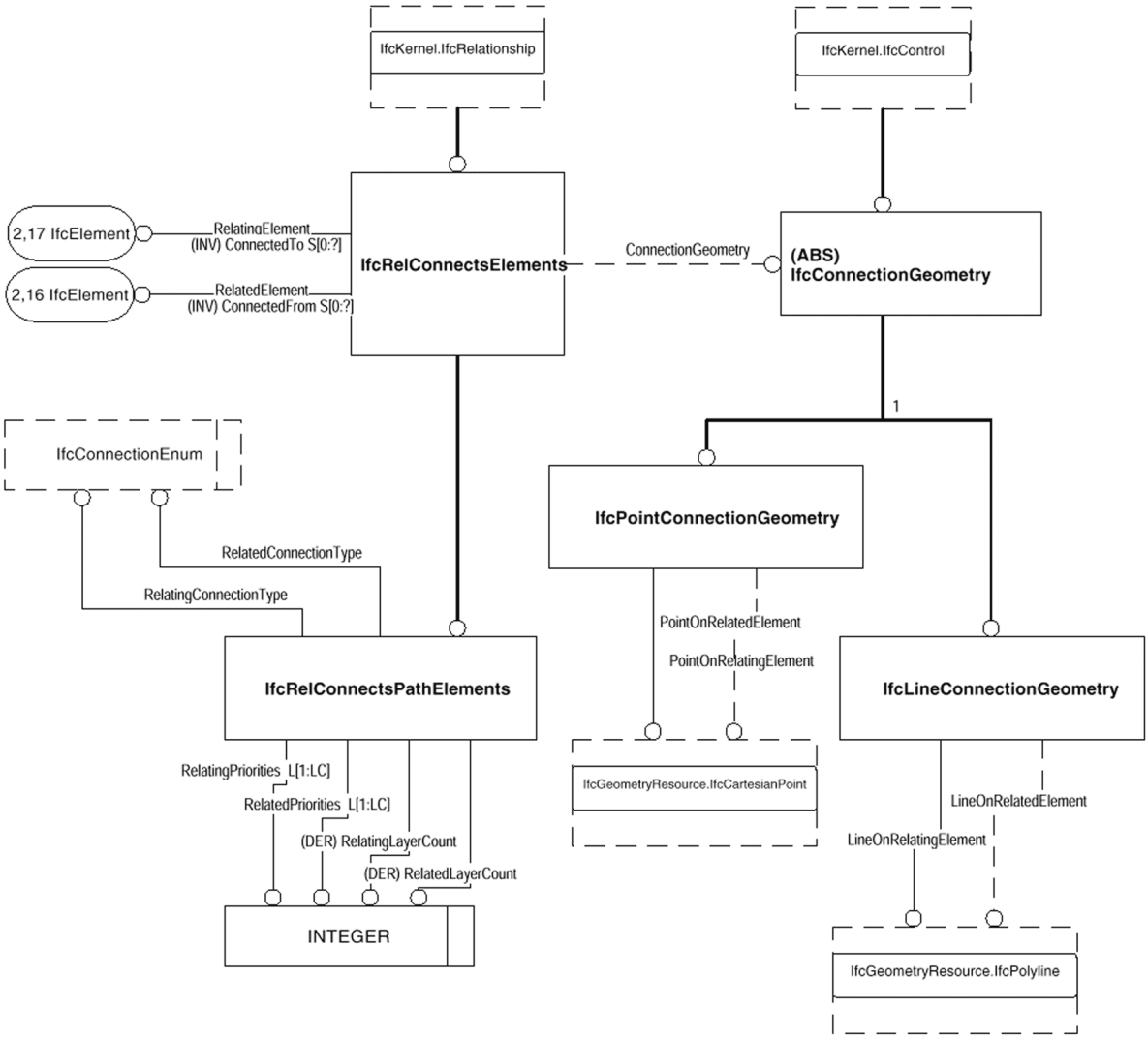
Page 3 du Schéma IfcProductExtension

Page 5 du Schéma IfcProductExtension



Page 5 du Schéma IfcProductExtension

Page 6 du Schéma IfcProductExtension

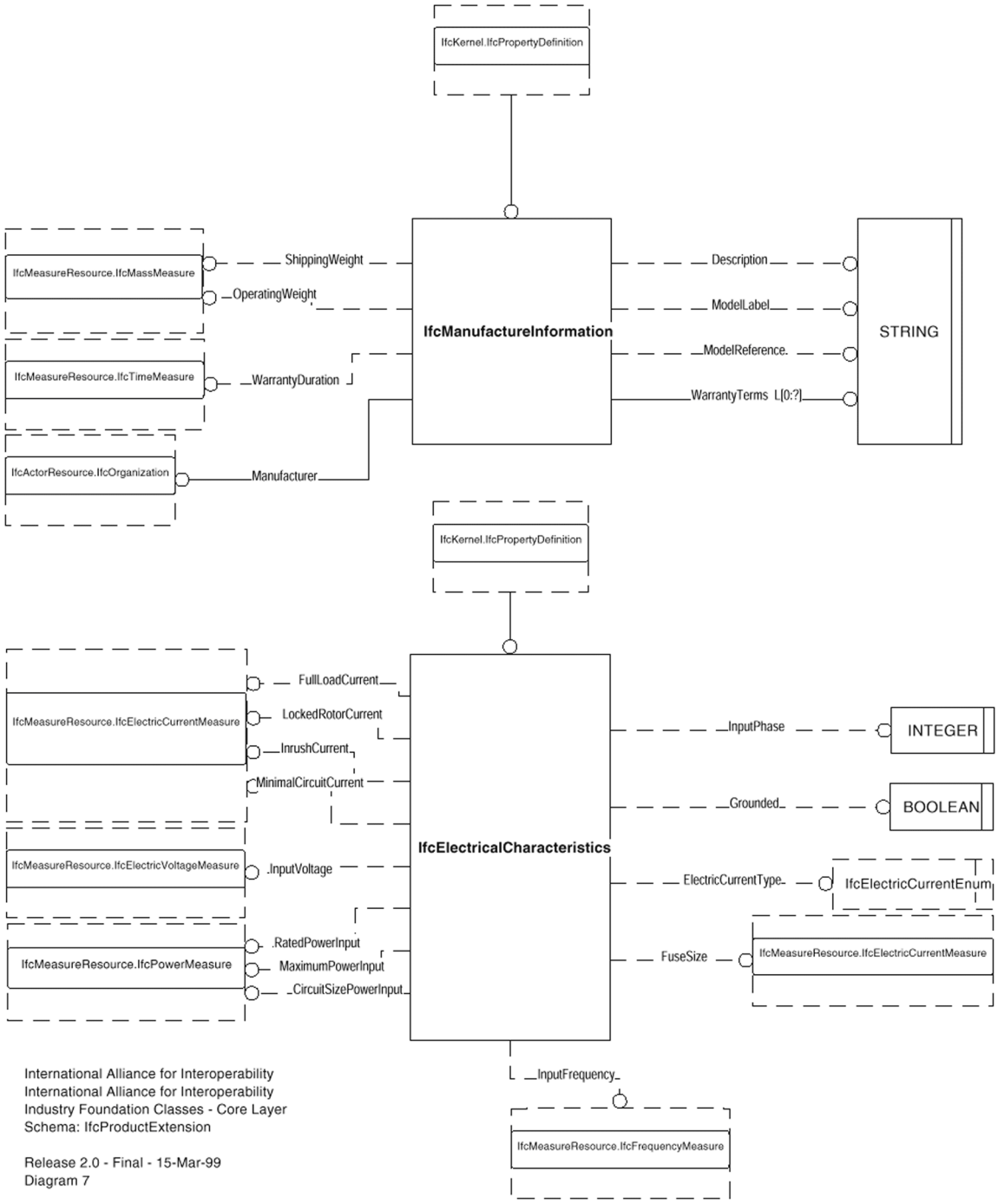


International Alliance for Interoperability
Industry Foundation Classes - Core Layer
Schema: IfcProductExtension

Release 2.0 - Final - 15-Mar-99
Diagram 6

Page 6 du Schéma IfcProductExtension

Page 7 du Schéma IfcProductExtension



Page 7 du Schéma IfcProductExtension.

Dictionnaire

Acteur :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Utilisateur d'un système d'information, mais qui peut aussi être modélisé comme un objet dans un modèle conceptuel

action :

Voir l'unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un modèle dynamique, action ponctuelle qui peut être exécutée à l'entrée, à la sortie ou pendant une transition

activité :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un modèle dynamique, action répétitive qui a une durée dans le temps, exécutée par un objet d'une classe

Attribut :

Voir l'unité "Le concept d'objets dans les logiciels de DAO, CAO et de calcul".

Même sens que "Propriété", c'est à dire description associée uniquement à un objet.

Booch Grady :

Auteur de méthodes de spécification formelle, qui a donné son nom à cette méthode. Co-auteur de la méthode UML

Classe :

Voir l'unité "L'IA et les IFC : introduction".

Dans un LOO, et dans les IFC, une classe regroupe des objets de même type, possédant des propriétés et un comportement semblable.

Concept :

Voir l'unité "Le concept d'objets dans les logiciels de DAO, CAO et de calcul".

Dans un modèle conceptuel, c'est une classe (unique) d'objets pertinente pour décrire le système d'information et son organisation sémantique.

concurrence :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un diagramme d'état d'un modèle dynamique, la transition est effectuée dès que l'un des sous-états en concurrence change d'état.

contexte :

Dans un diagramme de collaboration du modèle dynamique, partie du modèle statique servant de support aux interactions

Contraintes :

Voir unité "Se comprendre autour d'un système d'information".

En spécification formelle, une contrainte s'applique pour filtrer la mise en relation entre les objets des deux concepts en présence.

diagramme d'états :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un modèle dynamique. Associé à une seule classe d'objets, représente tous ses états possibles et les transitions entre ces états.

diagramme de collaboration :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Synthèse réalisée à partir du modèle statique et des diagrammes dynamiques qui collaborent pour offrir une vue du système limitée à une fonctions.

diagramme Scénario :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Formalise un Use-case, constitué d'évènements mettant en jeu des objets de plusieurs classes, et cités dans un ordre chronologique.

Etat :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un modèle dynamique, se dit d'un objet dont les propriétés sont constantes entre deux évènements

Evénement :

Voir l'unité "Quelques exemples d'élaboration de modèles conceptuels".

Action brusque qui n'a pas de durée, déclenchée par toute cause intérieure ou extérieure au système d'information.

EXPRESS :

Langage formel normalisé, pour décrire la structure de bases de données orientées objets. EXPRESS est un outil de STEP.

Le C.S.T.B. a développé un traducteur de schémas NIAM, qui produit des instructions EXPRESS.

EXPRESS-G :

Formalisme graphique pour décrire une structure de base de données, comme pour NIAM. L'avantage d'EXPRESS-G sur NIAM est qu'il constitue un outil de STEP, qui permet immédiatement d'obtenir une traduction de la base de données, en langage EXPRESS.

Inconvénient : il est moins pédagogique que NIAM. EXPRESS-G est donc réservé aux informaticiens

gardiens :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un diagramme d'état d'un modèle dynamique, conditions si elles existent pour qu'une transition s'accomplisse

granulométrie :

Voir l'unité "Quelques exemples d'élaboration de modèles conceptuels".

Finesse d'observation des éléments du modèle.

Héritage :

Voir l'unité "Le concept d'objet dans les logiciels de DAO, CAO et de calcul".

Dans un LOO, et dans les IFC, mécanisme qui permet à des objets d'une sous-classe de bénéficier des propriétés des classes "parents"

'homogénéité :

Voir l'unité "Comprendre et échanger les vues métiers du bâtiment".

Dans un modèle, propriété d'une paroi : en tout point de sa surface, les propriétés géométriques, fonctionnelles et physiques sont constantes.

Idée :

Voir l'unité "L'IAI et les IFC : introduction".

Terme utilisé dans la méthode de spécification NIAM : information mettant en jeu une seule relation (ensembliste) entre deux concepts.

Jacobson Ivar :

Auteur de méthodes de spécification formelle, inventeur des « Case-Use », co-auteur de la méthode UML.

messages :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un langage à objets ou un modèle dynamique : Action brusque qui n'a pas de durée, émise uniquement par un objet d'une classe

méta-modèle :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Ensemble des règles syntaxiques du langage formel de description d'un modèle (conceptuel)

modèle conceptuel :

Ce terme est développé dans l'unité "Le concept d'objet dans les logiciels de DAO, CAO et de calcul".

Description formelle des concepts véhiculés dans un modèle de base de données, focalisée sur l'aspect sémantique (statique) du système d'information.

modèle dynamique :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Un des trois aspects de la description d'un modèle conceptuel, qui prend en compte le contexte temporel de l'utilisation du système d'information.

NIAM :

"Nijssen Information Analysis Method". Méthode de spécification formelle de données, utilisable dans n'importe quel domaine pour décrire sans ambiguïté une organisation de concepts, d'objets, y compris les relations et attributs associés. Cette méthode est normalisée (STEP et ISO en 1983)

objets :

Voir l'unité "Le concept d'objet dans les logiciels de DAO, CAO et de calcul".

Nomme indifféremment un type d'objet, ou une occurrence de la classe.

Voir "Orienté Objet" et "Occurence (ou instance)".

OMT :

Object Modeling Technique : Méthode de specification formelle de James Rumbaugh.

OOSE :

Object Oriented Software Engineering : Méthode de spécification formelle de Ivar Jacobson

Opération :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un modèle dynamique, procédures déclenchées par l'objet qui change d'état, et qui se décomposent en actions et activités

Rôles :

En spécification formelle, et surtout dans NIAM, un rôle formalise l'expression d'une relation orientée entre deux concepts.

Rumbaugh James :

Auteur de méthodes de spécification formelle, co-auteur de la méthode UML.

synchronisation :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un diagramme d'état d'un modèle dynamique, la transition est effectuée si les deux sous-états en synchronisation changent d'état

systèmes d'information :

Voir unité "Pourquoi inventer des modèles pour la construction ?".

Ensemble des logiciels et des acteurs qui exploitent la connaissance modélisée d'un objet complexe, réduit à différents points de vue.

Totalité :

Voir unité "Se comprendre autour d'un système d'information".

Indique que dans une relation orientée tous les objets de l'ensemble de départ sont concernés. Se traduit par "Tout objet" ou "Chaque objet".

Traçabilité :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Conserver la continuité du cheminement de la transformation d'une information dans le développement d'un modèle, à toutes ses étapes

Transition :

Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Dans un modèle dynamique, décrit la procédure d'un changement d'état. Voir diagramme d'états.

UML :

Unified Modeling Language, méthode de spécification formelle résultat d'une synthèse entre les trois méthodes OMT, Booch et OOSE

Unicité :

Indique que dans une relation orientée les objets de l'ensemble de départ ne sont liés qu'à un seul objet de l'ensemble d'arrivée.

Use case :

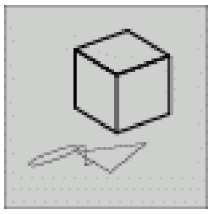
Voir unité "Quelques exemples d'élaboration de modèles conceptuels".

Cas d'utilisation, en français : Liste des actions à exécuter par le système d'information, imaginées par les acteurs pour définir une fonction.

Concepts IFC

Référence 1

Elément

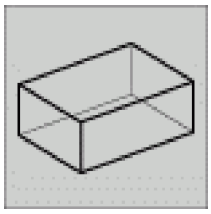


Tout "composant" matériel du "bâtiment", plein ou vide comme une "ouverture", à l'exclusion des espaces et "local".

ATTRIBUTS : caractérisé par les attributs du "produit", avec en plus les relations de voisinage : connexions topologiques ou de contact, appartenances, ces relations pouvant être définies par l'utilisateur selon les performances du logiciel.

Référence 2

Elément spatial



Concept abstrait pour définir un élément spatial composé soit d'un espace mesurable (possédant une surface et un volume, c'est à dire un "local"), soit des limites (frontières) de l'espace (c'est à dire la face visible des murs, planchers et sous faces de plafonds du local, ou "nu de local"). Ces deux types d'éléments doivent obligatoirement admettre une représentation géométrique.

Pas d'ATTRIBUTS à ce niveau.

Référence 3

IfcBuildingElement

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 4

IfcKernel

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 5

IfcProductExtension

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 6

IfcRelAssemblesSpaces

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 7

IfcRelationShip

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 8

IfcSpace

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 9

IfcSpaceBoundary

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

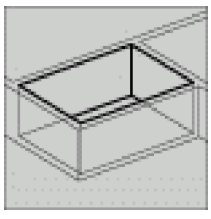
Référence 10

IfcSpatialElement

Voir, en fichier annexe, le fichier imprimable "Lexique Anglais Français des objets et types définis IFC utilisés dans l'arborescence d'héritage".

Référence 11

Local

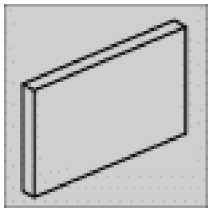


En français, le terme local est plus précis que celui d'espace et correspond à la définition sémantique de « IfcSpace ». Le local est entièrement délimité par des "composants" réels (murs, planchers, plafonds) et/ou des frontières virtuelles. Les IFC définissent toute une série de hauteurs et d'altitudes pour un local vu en coupe verticale (voir paragraphe « Le concept de local et son environnement dans les IFC).

ATTRIBUTS : liste des "nu de local" composants les frontières matérielles ou virtuelles du local (voir plus loin la définition du "nu de local"), type d'espace : intérieur ou extérieur (dans ce cas, les frontières virtuelles peuvent être majoritaires), nom court (de référence), nom complet (long), périmètre, surface, volume, Hauteur brute sous plafond (revêtement de sol et de plafond, même avec faux plafonds, ignorés), hauteur d'étage (de sol brut du plancher bas au sol brut du plancher haut), dans les deux cas, une moyenne est admise si les locaux ne sont pas prismatiques, hauteur nette sous plafond (entre les nus finis du sol et du plafond), altitude du nu fini du plancher, liste éventuelle des locaux élémentaires qui composent le présent local, local du niveau supérieur auquel appartient le présent espace, et liste des locaux du même groupe.

Référence 12

Mur

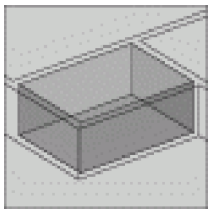


Tout type de mur, toutes fonctions, sauf les "cloison visuelle" et "façade rideaux". Sa géométrie peut être simple (rectangulaire) ou plus complexe (mur à rive supérieure en ligne brisée).

ATTRIBUTS : information des couches du mur (épaisseurs, constituants hormis les revêtements, matériaux), surface, volume, et relations avec les autres murs suivants et précédents. L'utilisateur ne décrit pas directement les relations entre murs, mais doit donner des priorités relatives entre types de murs (pour permettre aux logiciels de redessiner correctement les jonctions).

Référence 13

Nu de local



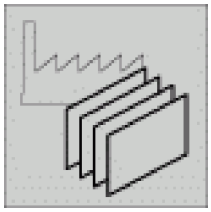
Traduction la plus appropriée de « IfcSpaceBoundary ». Définition sémantique des IFC : morceau de frontière, surface de délimitation du local en regard de chaque local adjacent, quelque soit son niveau. Cette définition est lourde de conséquences (voir paragraphe « Le concept de local et son environnement dans les IFC).

ATTRIBUTS : Type de nu (physique ou virtuel, contre l'extérieur ou adjacent à un local extérieur), surface du nu, référence aux deux locaux séparés par le nu, dans le cas ou

un local adjacent existe (on peut considérer l'extérieur comme une sorte de local).

Référence 14

Produit

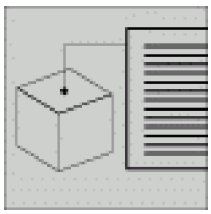


Classe abstraite pour représenter tout produit manufacturé, découpé ou fabriqué in situ, ou encore résultat de frontières matérielles comme les espaces ("local").

ATTRIBUTS : localisation, représentation géométrique ou forme associée, liste de spécifications

Référence 15

Ressource



Ne pas confondre avec la couche indépendante d'objets regroupés sous le nom de « Ressources » dans l'architecture du modèle IFC, utilisée par tous les objets du modèle du projet. Il s'agit ici exclusivement des informations nécessaires aux "procédures" (données nécessaires aux logiciels de calcul) concernant les études de prix, de planning et d'ordonnancement de travaux. Il existe déjà dans le modèle IFC de nombreux objets renseignant les "produits" et "procédures" pour un traitement des coûts et plannings. Cette présente classe (et ses sous-classes) est destinée à servir les objets qui ne sont pas normalisés par les IFC (et qui sont décrits comme des "objets utilisateur", par exemple, comme les moyens de levage, les banches de coffrage, les moyens de transport ...). Cette classe d'objets Ressource peut aussi servir à compléter la description des ressources des objets modélisés par les IFC, la liaison s'effectuant par des relations explicites.

ATTRIBUTS : Description sémantique du type de ressource (option), code de référence du type de ressource, nom, référence de codification (option), unité de mesure de la ressource (option).