

MEGA for UML

Guide d'utilisation



HOPEX V1R2-R3
1ère édition (juillet 2015)

Les informations contenues dans ce document pourront faire l'objet de modifications sans préavis et ne sauraient en aucune manière constituer un engagement de la société MEGA International.

Aucune partie de la présente publication ne peut être reproduite, enregistrée, traduite ou transmise, sous quelque forme et par quelque moyen que ce soit, sans un accord préalable écrit de MEGA International.

© MEGA International, Paris, 1996 - 2015

Tous droits réservés.

MEGA for UML et MEGA sont des marques réservées de MEGA International.

Windows est une marque réservée de Microsoft.

Les autres marques citées appartiennent à leurs propriétaires respectifs.

SOMMAIRE



| | |
|---------------------------|----------|
| Sommaire | 3 |
|---------------------------|----------|

| | |
|---|----------|
| A propos de MEGA for UML | 9 |
|---|----------|

| | |
|--|-----------|
| Conventions utilisées dans le guide | 10 |
|--|-----------|

| | |
|-------------------------------|-----------|
| Présentation | 11 |
|-------------------------------|-----------|

- Analyser les cas d'utilisation* 11
- Identifier les objets* 11
- Décrire les comportements* 11
- Représenter les interactions entre les objets* 11
- Répartir les classes entre les paquetages* 12
- Définir les interfaces* 12
- Spécifier la mise en oeuvre* 12

| | |
|--|-----------|
| Organisation des diagrammes d'UML | 13 |
|--|-----------|

- Organisation générale* 13
- Spécification détaillée* 13
- Spécification technique et mise en oeuvre* 14
- Points d'entrée des diagrammes d'UML* 14

| | |
|--------------------------------------|-----------|
| Lancer MEGA for UML | 15 |
|--------------------------------------|-----------|

- Découvrir l'espace de travail MEGA for UML 17
-

| | |
|--|-----------|
| Le diagramme de cas d'utilisation | 19 |
|--|-----------|

| | |
|--|-----------|
| Créer un diagramme de cas d'utilisation | 20 |
|--|-----------|

- Créer un paquetage 20
- Créer le diagramme de cas d'utilisation du paquetage 20
- Ouvrir un diagramme de cas d'utilisation 21

| | |
|---|-----------|
| Les éléments du diagramme de cas d'utilisation | 22 |
|---|-----------|

- Les acteurs 22

| | |
|---|----|
| Les cas d'utilisation | 23 |
| <i>Faire un zoom sur un cas d'utilisation</i> | 23 |
| Les paquetages | 24 |
| Les participations | 25 |
| <i>Exemple de participation</i> 26 | |
| <i>Créer une participation</i> | 26 |
| <i>Multiplicités d'une participation</i> | 27 |
| Les associations entre cas d'utilisation : extension et inclusion | 27 |
| <i>Relation d'inclusion</i> | 27 |
| <i>Relation d'extension</i> | 28 |
| Les généralisations | 30 |
| Les interfaces | 32 |

Le diagramme de classes 35

| | |
|---|-----------|
| Présentation du diagramme de classes | 36 |
| <i>Le diagramme de classes : synthèse</i> | 37 |
| Créer un diagramme de classes | 38 |
| Les classes | 39 |
| Définition d'une classe | 39 |
| Créer une classe | 40 |
| <i>Retrouver une classe existante</i> | 40 |
| Propriétés d'une classe | 41 |
| <i>Onglet Caractéristiques</i> 42 | |
| <i>Onglet Général</i> 43 | |
| <i>Onglet Textes</i> 43 | |
| <i>Autres onglets</i> 43 | |
| Stéréotype d'une classe | 43 |
| <i>Option d'affichage des stéréotypes</i> | 45 |
| Les attributs | 46 |
| Définition d'un attribut | 46 |
| Spécifier les attributs d'une classe | 46 |
| <i>Attributs hérités</i> | 48 |
| Propriétés des attributs | 49 |
| <i>Types</i> | 50 |
| Les opérations | 51 |
| Définition d'une opération | 51 |
| Spécifier les opérations d'une classe | 51 |
| <i>Opérations héritées</i> | 52 |
| Propriétés d'une opération | 52 |
| Signature d'une opération ou d'un signal | 53 |
| <i>Syntaxe des signatures</i> | 54 |
| Paramètres d'une opération | 55 |
| Méthodes d'une opération (comportement opaque) | 55 |
| Conditions d'une opération | 56 |
| Exceptions d'une opération | 56 |
| Afficher les attributs et les opérations d'une classe | 57 |
| Les signaux | 58 |
| Définition d'un signal | 58 |

| | |
|--|-----------|
| Spécifier les signaux d'une classe | 58 |
| <i>Créer un signal émis ou reçu</i> | 58 |
| <i>Propriétés d'un signal</i> | 59 |
| <i>Paramètres d'un signal</i> | 60 |
| Les associations | 61 |
| Créer une association | 62 |
| Les rôles des associations | 62 |
| Multiplicité d'un rôle | 63 |
| <i>Préciser la multiplicité d'un rôle</i> | 65 |
| Navigabilité d'un rôle | 66 |
| <i>Préciser la navigabilité d'un rôle</i> | 66 |
| Agrégation d'un rôle | 67 |
| <i>Préciser l'agrégation d'un rôle</i> | 67 |
| Composition d'un rôle | 67 |
| Rôle modifiable | 68 |
| Ordre d'un rôle | 69 |
| Propriété statique d'un rôle | 69 |
| Qualificatif d'un rôle | 70 |
| Surcharger un rôle | 70 |
| Les classes d'association | 71 |
| Définir une association "plus que binaire" | 72 |
| Les associations réflexives | 72 |
| <i>Créer une association réflexive</i> | 73 |
| Les généralisations | 74 |
| Qu'est-ce qu'une généralisation | 74 |
| <i>Exemple 75</i> | |
| Cas de plusieurs sous-classes | 76 |
| Intérêt des sous-classes | 76 |
| Héritage Multiple | 77 |
| Créer une généralisation | 77 |
| Discriminant | 78 |
| Spécifier les interfaces | 79 |
| Créer une interface | 79 |
| Spécifier les dépendances | 81 |
| Spécifier des classes paramétrées | 82 |
| Spécifier une classe paramétrée | 82 |
| Les contraintes | 84 |
| Le diagramme d'objets | 85 |
| Les objets | 85 |
| <i>Créer un objet (une instance) 86</i> | |
| <i>Propriétés d'une instance 86</i> | |
| <i>Valeur d'un attribut 87</i> | |
| Les liens | 87 |
| <i>Créer un lien 88</i> | |
| <i>Propriétés d'un lien 88</i> | |
| <i>Propriétés d'un rôle 88</i> | |
| L'éditeur de classes | 90 |
| Ouvrir l'éditeur de classes | 90 |
| Paramètres d'affichage de l'éditeur de classes | 90 |
| Propriétés des objets | 91 |
| Créer des objets dans l'éditeur de classes | 92 |

| | |
|--|-----------|
| <i>Créer une association entre deux classes</i> | 93 |
| <i>Créer une généralisation entre deux classes</i> | 93 |
| Générer un diagramme de classes | 94 |
| Fonctionnalité de réorganisation automatique. | 94 |

Les diagrammes de structure et de déploiement. 97

| | |
|---|------------|
| Le diagramme de paquetages | 98 |
| Créer un diagramme de paquetages | 98 |
| Les paquetages | 99 |
| <i>Retrouver un paquetage existant</i> | 99 |
| Les classes. | 100 |
| <i>Retrouver les classes existantes</i> | 100 |
| Spécifier les dépendances dans un diagramme de paquetage. | 100 |
| Le diagramme de composants | 101 |
| Les composants | 101 |
| Les interfaces. | 102 |
| <i>Créer les interfaces des composants</i> | 102 |
| <i>Relier les interfaces aux autres objets</i> | 102 |
| <i>Relier des interfaces</i> | 103 |
| Les ports | 103 |
| <i>Associer une interface à un port</i> | 103 |
| Les connecteurs | 103 |
| <i>Connecteur de délégation</i> | 104 |
| <i>Connecteur d'assemblage</i> | 104 |
| Le diagramme de structure composite | 106 |
| Les parties | 106 |
| <i>Multiplicité</i> | 107 |
| Les collaborations | 107 |
| <i>Utilisation de collaboration</i> | 107 |
| Les liens de dépendance | 108 |
| <i>Nature de la dépendance</i> | 109 |

Le diagramme de machine à états. 111

| | |
|---|------------|
| Présentation du diagramme de machine à états | 112 |
| <i>Exemple de diagramme de machine à d'états</i> | 112 |
| Créer un diagramme de machine à états | 112 |
| Les états | 113 |
| <i>Exemples d'état d'objet</i> : | 113 |
| Créer un état | 113 |
| <i>Les types d'état</i> | 113 |
| <i>Pseudo-états</i> | 114 |
| Précision comportementale d'un état. | 115 |
| Propriétés d'un état. | 116 |

| | |
|---|------------|
| Les transitions entre états | 117 |
| Créer une transition | 117 |
| Les types de transition | 117 |
| <i>Transition externe</i> | 117 |
| <i>Transition interne</i> | 117 |
| <i>Transition locale</i> | 118 |
| Effet d'une transition | 118 |
| <i>Affichage des effets d'une transition.</i> | 118 |
| Événement déclencheur d'une transition | 118 |

Le diagramme d'activités 121

| | |
|--|------------|
| Présentation du diagramme d'activités | 122 |
| Créer un diagramme d'activités | 122 |
| Les partitions | 123 |
| Créer une partition | 123 |
| Propriétés d'une partition | 124 |
| Les noeuds | 125 |
| Les actions | 125 |
| <i>Créer une action</i> | 125 |
| <i>Types d'action</i> | 125 |
| Noeuds de paramétrage | 125 |
| Noeuds de contrôle | 126 |
| <i>Types de noeud de contrôle</i> | 126 |
| Pins d'entrée, de sortie et d'échange | 127 |
| <i>Pin d'entrée</i> | 127 |
| <i>Pin de sortie</i> | 127 |
| <i>Pin d'échange</i> | 127 |
| Les Flux. | 128 |
| <i>Flux de contrôle.</i> | 128 |
| <i>Flux d'objets</i> | 128 |

Les diagrammes d'interaction 129

| | |
|---|------------|
| Les interactions | 130 |
| Créer une interaction | 130 |
| Créer un diagramme d'interaction | 130 |
| Le diagramme de séquence | 131 |
| <i>Exemple de diagramme de séquence</i> | 131 |
| Les lignes de vie | 132 |
| <i>Créer une ligne de vie</i> | 132 |
| <i>Propriétés d'une ligne de vie</i> | 132 |
| Les messages | 133 |
| <i>Créer un message</i> | 134 |
| <i>Types de messages</i> | 134 |
| Occurrence d'exécution | 135 |

| | |
|---|------------|
| <i>Créer une occurrence d'exécution</i> | 135 |
| Occurrence d'événement | 135 |
| <i>Calcul des numéros de séquence</i> | 136 |
| Fragment combiné | 137 |
| <i>Créer un fragment combiné</i> | 137 |
| <i>Type d'opérateur d'interaction</i> | 138 |
| <i>Opérande d'interaction</i> | 140 |
| Utilisation d'interaction | 141 |
| Porte | 142 |
| Continuation | 142 |
| Le diagramme de communication | 143 |
| <i>Exemple 143</i> | |
| <i>Objets du diagramme</i> | 144 |
| Le diagramme de vue générale d'interaction | 145 |

Le diagramme de déploiement 147

| | |
|---|------------|
| Présentation du diagramme de déploiement | 148 |
| Objets du diagramme de déploiement | 148 |
| <i>Noeud</i> | 148 |
| <i>Chemin de communication</i> | 149 |
| <i>Composant</i> | 149 |
| <i>Artefact</i> | 149 |
| <i>Manifestation</i> | 149 |
| <i>Spécification de déploiement</i> | 149 |
| <i>Configuration</i> | 150 |

Glossaire 151

Annexe : Type des attributs 157

| | |
|---|------------|
| Types élémentaires | 158 |
| Définir un type élémentaire | 158 |
| Paquetages et types élémentaires | 160 |
| Paquetages | 160 |
| Définir de nouveaux types élémentaires | 163 |
| Type élémentaire composé | 164 |

Index 165

A PROPOS DE MEGA FOR UML








Le langage de modélisation UML (Unified Modeling Language) s'est érigé comme standard pour la modélisation graphique des systèmes d'information. MEGA for UML offre un ensemble de vues et d'outils vous permettant de modéliser votre SI conformément à la version 2.3 de ce standard.

Ce guide a pour objectif de vous faire découvrir les principales fonctionnalités de MEGA for UML.

- ✓ ["Conventions utilisées dans le guide", page 10](#)
- ✓ ["Présentation", page 11](#)
- ✓ ["Organisation des diagrammes d'UML", page 13](#)
- ✓ ["Lancer MEGA for UML", page 15](#)

CONVENTIONS UTILISÉES DANS LE GUIDE

-  *Remarque sur les points qui précèdent.*
-  *Définition des termes employés.*
-  *Astuce qui peut faciliter la vie de l'utilisateur.*
-  *Compatibilité avec les versions précédentes.*
-  **Ce qu'il faut éviter de faire.**



Remarque très importante à prendre en compte pour ne pas commettre d'erreurs durant une manipulation.

Les commandes sont présentées ainsi : **Fichier > Ouvrir**.

Les noms de produits et de modules techniques sont présentés ainsi : **MEGA**.

PRÉSENTATION

MEGA for UML vous permet de :

Analyser les cas d'utilisation

Une réflexion sur les fonctionnalités attendues du futur système est nécessaire avant sa conception. Les composants de ce système vont en effet être utilisés par les acteurs de l'organisation pour effectuer leur mission. Les divers "cas d'utilisation" de ce système vont être présentés dans les **diagrammes de cas d'utilisation**.

Ils serviront de point de départ pour la découverte des objets.

Ils permettront ensuite de valider l'utilisation de ces objets à l'aide des diagrammes d'interaction.

Ils donneront enfin des critères de regroupement en "paquetage" pour les objets découverts.

Identifier les objets

Les objets ayant une structure semblable, le même comportement et les mêmes types de relations avec d'autres objets sont réunis dans une classe.

Le **diagramme de classes** permet d'identifier les objets mis en jeu à l'intérieur du système et de définir leur structure en termes d'attributs et d'opérations ainsi que les relations entre eux. Le **diagramme d'objets** montre les instances compatibles avec un diagramme de classes particulier qu'on peut ainsi valider avec un exemple.

Décrire les comportements

Le **diagramme de machine à états** permet de définir le comportement d'un objet vis-à-vis des sollicitations internes ou externes auxquelles il peut être soumis. Chacun des états dans lequel peut se trouver l'objet est indiqué ainsi que les réactions de l'objet à un événement donné en fonction de l'état où il se trouve.

Le **diagrammes d'activité** décrit également un comportement, mais en termes d'actions.

Représenter les interactions entre les objets

Le dialogue qui s'instaure entre les différents objets concernés par l'événement pour y répondre peut être représenté dans des **diagrammes d'interaction**.

Les diagrammes d'interaction insistent sur les échanges qui ont lieu entre les objets.

Le **diagramme de séquence** présente ces mêmes échanges en mettant en évidence leur chronologie.

Le **diagramme de communication** met l'accent sur l'organisation structurelle des objets qui envoient et reçoivent des messages.

Le **diagramme de vue générale d'interaction** donne une vue d'ensemble du flot de contrôle.

Répartir les classes entre les paquetages

Une fois les objets identifiés, il est aisé de répartir les classes qui les implémentent entre les différents paquetages. Les regroupements de ces classes effectués dans le **diagramme de paquetages** sont faits de manière à minimiser les échanges entre les différents paquetages. Ils obéissent à deux critères : l'un, technique, lié à leur environnement d'exécution et l'autre, organisationnel, lié à l'emploi qui en sera fait par les différents utilisateurs à équiper pour chaque cas d'utilisation.

Définir les interfaces

Pour respecter le principe d'encapsulation, la répartition des éléments entre les composants est stricte. Il est donc nécessaire de prévoir les interfaces entre les éléments ayant des relations entre eux et qui appartiennent cependant à des composants différents.

Le **diagramme de composants** et le **diagramme de structure composite** présentent l'interdépendance des composants ou éléments d'un composant.

La définition des interfaces des objets ainsi que l'adhésion à un protocole d'échange normalisé (CORBA2, DCOM/OLE) sont des facteurs clés de l'interopérabilité, c'est-à-dire la capacité à faire coopérer les objets développés et exploités dans des environnements hétérogènes.

Spécifier la mise en oeuvre

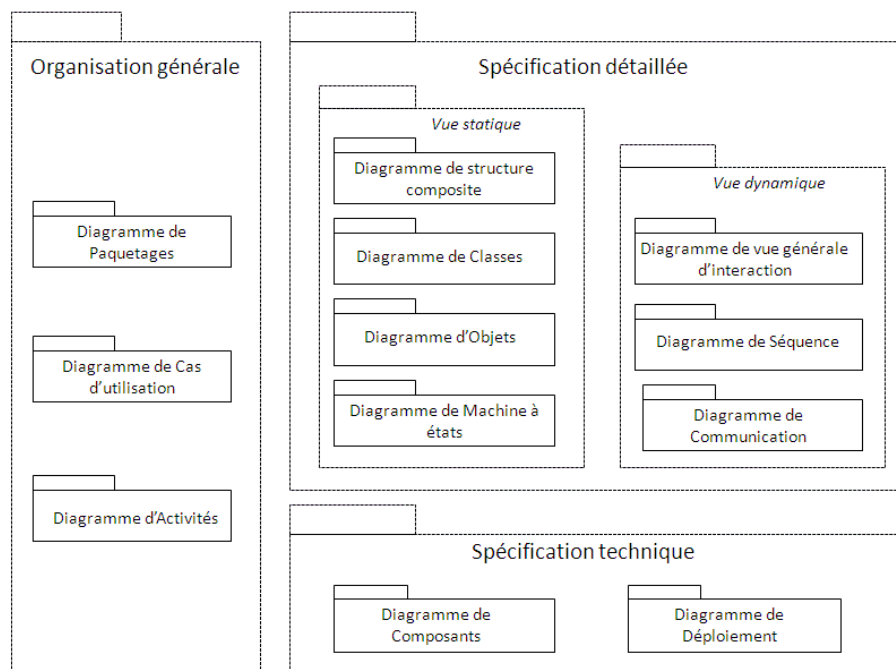
La mise en oeuvre des objets dans un environnement de travail concret peut être spécifiée les **diagrammes de déploiement**.

ORGANISATION DES DIAGRAMMES D'UML

Organisation générale

Les **diagrammes de cas d'utilisation** vous permettent de montrer les principales interactions entre le système étudié et son environnement, et de mettre en évidence ses principaux sous-systèmes.

Les **diagrammes de paquetages** constituent un découpage du système. Le découpage du système en paquetages est relativement structurant dans la mesure où un objet ne peut être que dans un seul paquetage. Vous pouvez commencer à dessiner des diagrammes de paquetages dès que vous avez identifié les principaux composants de votre système (Commercial, Production, Facturation, ...).



Spécification détaillée

Le diagramme principal est le **diagramme de classes**. Il décrit l'essentiel de la sémantique des objets composant le système. C'est le diagramme sur lequel les concepteurs passeront généralement le plus de temps. La découverte des classes se fait généralement par des aller-retour entre les diagrammes de classes et les diagrammes de séquence.

Le **diagramme de machine à états** permet de décrire l'aspect statique d'un objet, c'est à dire les différents états dans lesquels il peut se trouver et les transitions possibles entre ces états. Il permet ainsi de compléter la description d'une classe.

Les **diagrammes d'interaction** permettent de spécifier l'aspect dynamique du système en montrant l'interaction des objets entre eux. Ils permettent en particulier de décrire de façon détaillée les différents scénarios de fonctionnement d'un cas d'utilisation. Le diagramme de séquence explicite plutôt le déroulement d'un scénario dans le temps, tandis que le diagramme de communication insiste plutôt sur l'interaction entre les objets.

Spécification technique et mise en oeuvre

Le **diagramme de composants** décrit les différents composants techniques d'une application et leurs interactions.

Le **diagramme de structure composite** précise les collaborations entre les composants ou éléments d'un composant dans l'exécution d'une tâche commune.

Le **diagramme de déploiement** permet de préciser l'architecture technique du système en indiquant sur quels postes de travail ou sur quels nœuds du système informatique seront installés les différents composants de l'application.

Points d'entrée des diagrammes d'UML


| Diagramme | Points d'entrée |
|---|---|
| Diagramme de classes | Paquetage, Classe, Cas d'utilisation |
| Diagramme d'objets | Classe, Composant, Paquetage, Cas d'utilisation |
| Diagramme de composants | Composant, Paquetage |
| Diagramme de structure composite | Composant, Classe, Collaboration |
| Diagramme de déploiement | Paquetage |
| Diagramme de paquetages | Paquetage, Bibliothèque |
| Diagramme de cas d'utilisation | Paquetage, Cas d'utilisation |
| Diagramme de séquence (UML2) | Interaction |
| Diagramme de communication | Interaction |
| Diagramme de vue générale d'interaction | Interaction |
| Diagramme d'activités (UML2) | Activité |
| Diagramme de machine à états | Machine à état, Machine à état de protocole |

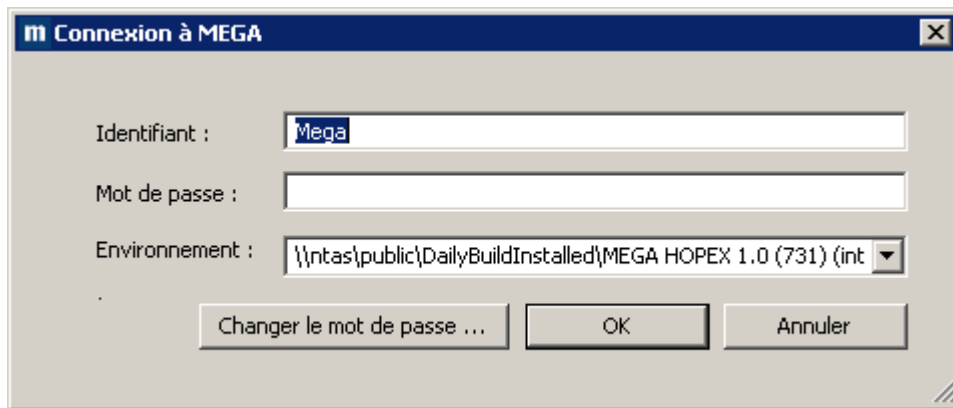
LANCER MEGA FOR UML

Pour lancer **MEGA** :

Pour faciliter votre première approche avec **MEGA**, ce guide est basé sur des exemples disponibles dans la Base "MEGA (Tutorial)" de l'Environnement "Démonstration". L'utilisateur "Mister Guide" a été installé pour vous guider dans votre découverte de **MEGA**.

Pour lancer **MEGA** :

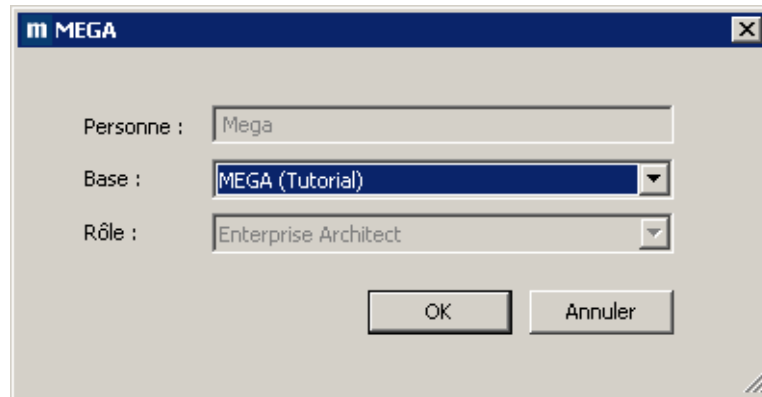
1. Double-cliquez sur l'icône **MEGA**  Mega.exe.
La fenêtre d'identification apparaît.



2. Dans le champ **Environnement**, sélectionnez votre environnement de travail.
 - ☛ Si vous n'avez accès qu'à un environnement, celui-ci est automatiquement pris en compte et le champ de sélection de l'environnement est grisé.
 - ☛ Pour en savoir plus sur les environnements et les utilisateurs, voir les chapitres **Gérer les environnements** et **Gérer les utilisateurs** du guide **MEGA Administration**.
3. Dans le champ **Identifiant**, saisissez votre identifiant.
 - ☛ Si vous avez choisi l'environnement "Démonstration", vous pouvez travailler avec l'identifiant "Mister Guide".
4. Dans le champ **Mot de passe**, saisissez votre mot de passe (si nécessaire).


5. Cliquez sur **OK**.

Lorsque vous êtes authentifié, la fenêtre de sélection d'un espace de travail apparaît.



Le champ **Personne** est défini automatiquement ; il indique le nom de la personne associée à l'identifiant défini dans la fenêtre de connexion.

6. Dans le champ **Base**, sélectionnez votre base de travail.

☛ Le menu déroulant  vous permet de faire apparaître la liste des bases disponibles dans l'environnement. Lorsque vous commencez à modéliser les données de votre entreprise, il est recommandé de créer une nouvelle base dans un nouvel environnement.

☛ Si vous n'avez accès qu'à une seule base, celle-ci est automatiquement prise en compte et le champ de sélection de la base est grisé.

7. Dans le champ **Rôle**, cliquez sur la flèche et sélectionnez le profil ou le rôle avec lequel vous voulez travailler.

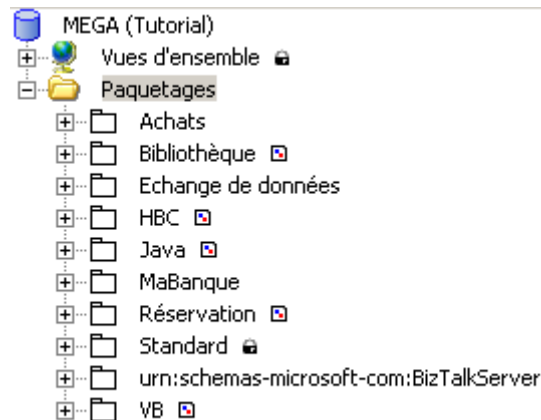
☛ Si vous n'avez qu'un rôle, celui-ci est automatiquement pris en compte.


8. Cliquez sur **OK**.

Votre espace de travail apparaît.

Découvrir l'espace de travail MEGA for UML

Dans le navigateur, cliquez sur le + devant "Paquetage" pour voir apparaître la liste des paquetages de premier niveau.



L'icône  visible à côté de certains paquetages signifie qu'ils sont décrits par un diagramme.

LE DIAGRAMME DE CAS D'UTILISATION



Le diagramme de cas d'utilisation constitue une première étape dans la description d'un système d'information. Il permet d'identifier les fonctionnalités que doit fournir le système afin de répondre aux besoins des acteurs de l'organisation ; il décrit en ce sens les interactions entre le système et les acteurs.

Les points suivants sont abordés ici :

- ✓ ["Créer un diagramme de cas d'utilisation", page 20](#)
- ✓ ["Les éléments du diagramme de cas d'utilisation", page 22](#)

CRÉER UN DIAGRAMME DE CAS D'UTILISATION

Un diagramme de cas d'utilisation permet de décrire les interactions entre les acteurs de l'organisation et le système dans chacun des *cas d'utilisation* envisagés.



Un cas d'utilisation est une suite d'actions qui amène un résultat observable pour un acteur particulier. Des scénarios illustrent les cas d'utilisation par l'exemple.

Ces cas d'utilisation sont regroupés dans des *paquetages* représentant les frontières du système.




Un paquetage partitionne le domaine d'étude et les travaux associés. Il permet de regrouper divers éléments, en particulier des cas d'utilisations et des classes. Un paquetage peut aussi contenir d'autres paquetages. Les paquetages sont liés entre eux à travers des rapports contractuels définissant leur interface.

Le paquetage est l'élément à partir duquel vous allez créer le diagramme de cas d'utilisation. Cependant, dans le cadre de systèmes complexes, vous pouvez également créer ce type de diagramme depuis un cas d'utilisation afin de détailler ce dernier.

Créer un paquetage

Pour créer un paquetage :

1. Ouvrez la fenêtre de navigation **Objets principaux**.
2. Cliquez avec le bouton droit sur le dossier **Paquetage** et sélectionnez **Nouveau > Paquetage**.
La fenêtre de création d'un paquetage s'ouvre.
3. Saisissez son **Nom**.
4. Indiquez éventuellement la bibliothèque ou le paquetage détenteur.
 *La bibliothèque par défaut est utilisée pour ranger un objet s'il n'y a pas de bibliothèque courante au moment de sa création.*
5. Cliquez sur **OK**.

Le paquetage est créé et ajouté dans la liste des paquetages.




*Quand le bouton **OK** ou **Créer** est grisé, c'est que la fenêtre où il apparaît n'a pas été complètement renseignée.*

Créer le diagramme de cas d'utilisation du paquetage

Pour créer un diagramme de cas d'utilisation :

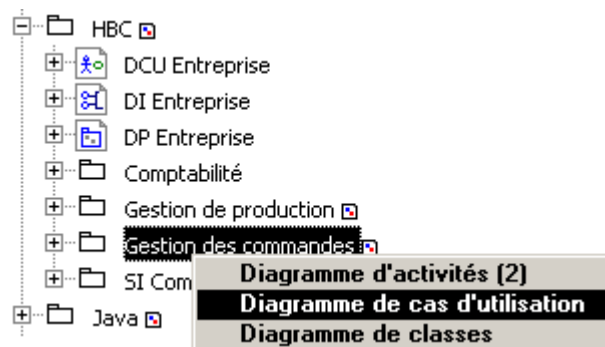
1. Cliquez avec le bouton droit sur le paquetage concerné.
2. Cliquez sur **Nouveau > Diagramme...**.
Une fenêtre apparaît avec les types de diagrammes possibles.
3. Sélectionnez le type de diagramme **Diagramme de cas d'utilisation**.

4. Vérifiez que l'option **Initialiser le diagramme** est cochée.
 L'option **Initialiser le diagramme** permet de prendre en compte l'environnement de l'objet dans le dessin.
5. Cliquez sur **Créer**.
Le diagramme créé s'ouvre dans la fenêtre d'édition de **MEGA**. Le cadre du paquetage est positionné à l'intérieur du dessin.

Ouvrir un diagramme de cas d'utilisation


Pour ouvrir un diagramme de cas d'utilisation existant :

1. Dans la fenêtre de navigation **Objets principaux**, cliquez avec le bouton droit sur le paquetage qui contient le cas d'utilisation.
2. Cliquez sur **Diagramme de cas d'utilisation**.




Le diagramme de cas d'utilisation s'ouvre.

Vous êtes dans l'outil de dessin de **MEGA for UML**. Les paragraphes qui suivent présentent les objets d'un diagramme de cas d'utilisation : des acteurs, des cas d'utilisation, des paquetages, etc., ainsi que des liens entre ces objets.

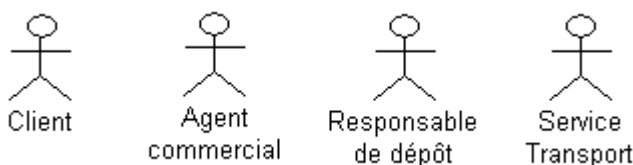
 Dans les exemples présentés dans ce guide, les noms des objets incluent, entre autres, des blancs, des majuscules et des caractères accentués. Lorsqu'il y a, pour un outil de génération qui utilise les spécifications effectuées avec **MEGA for UML**, des restrictions sur les caractères autorisés et sur la longueur des noms, il est préférable de se conformer à ces règles dès la spécification avec **MEGA for UML**.

LES ÉLÉMENTS DU DIAGRAMME DE CAS D'UTILISATION


Les acteurs


 Un acteur représente le rôle joué par quelque chose ou quelqu'un se trouvant dans l'environnement de l'entreprise ou du système étudié. Il est en relation avec le métier de l'entreprise et interagit avec le système dans différents cas d'utilisation. Ce peut être un élément de la structure de l'entreprise tel qu'une direction, un service ou un poste de travail.

Exemples d'acteurs :





Pour créer un *acteur* dans un diagramme :

1. Dans la barre d'objets, cliquez sur le bouton **Acteur (UML)** .
2. Cliquez sur le plan de travail du diagramme. La fenêtre **Ajout d'un acteur (UML)** s'ouvre.
3. Saisissez le nom de l'acteur, "Standardiste", par exemple.
4. Cliquez sur le bouton **Créer**.

 Lorsque le nom que vous avez saisi correspond à un acteur qui existe déjà dans la base, le libellé du bouton **Créer** devient **Relier**.

L'acteur apparaît alors dans le diagramme.

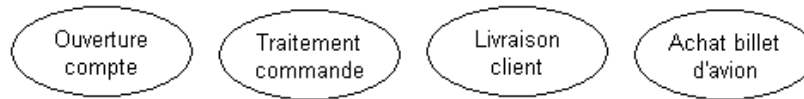
😊 Vous pouvez créer plusieurs éléments à la suite sans revenir à la barre d'outils en effectuant un double-clic sur le bouton .

😊 Pour revenir ensuite au mode normal, utiliser la touche <Echap>, ou cliquez sur un autre bouton de la barre d'outils, par exemple sur la flèche .


😊 Vous pouvez affecter la même taille à plusieurs éléments en leur appliquant la commande **Retailler à l'identique** du menu **Dessin**, après les avoir sélectionnés. Vous pouvez de même les aligner à l'aide de la commande **Aligner** de ce même menu. (Le dernier élément sélectionné est pris comme référence. Pour sélectionner plusieurs éléments, cliquez successivement sur chaque élément en maintenant la touche <Majuscules> enfoncée.)

Les cas d'utilisation

Exemples de *cas d'utilisation* : traitement d'une commande, livraison d'un client, ouverture d'un compte, envoi d'une facture, établissement d'un crédit, achat d'un billet d'avion, etc.



Pour créer un cas d'utilisation dans un diagramme :

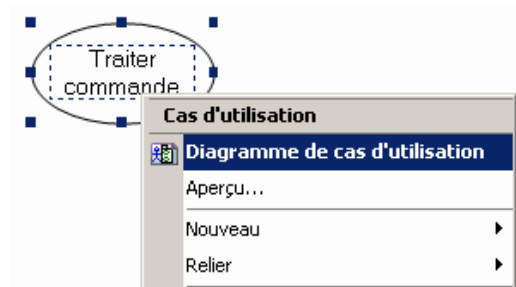
1. Cliquez sur le bouton **Cas d'utilisation**  dans la barre d'objets. La fenêtre **Ajout d'un cas d'utilisation** s'ouvre.
2. Saisissez le **Nom** du cas d'utilisation et cliquez sur le bouton **Créer**.

Le cas d'utilisation apparaît sur le diagramme.

Faire un zoom sur un cas d'utilisation

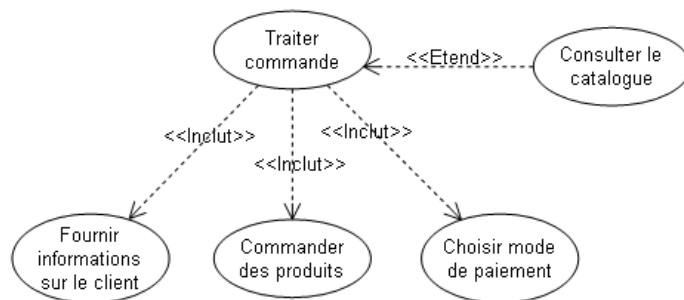
Pour ouvrir directement, depuis le diagramme du packaging, le diagramme qui décrit un cas d'utilisation :

1. Cliquez avec le bouton droit sur le cas d'utilisation.
2. Sélectionnez **Diagramme de cas d'utilisation**.



☛ Vous pouvez également créer un diagramme de classes ou un diagramme d'objets à l'aide de la commande Nouveau.

Le diagramme de cas d'utilisation s'ouvre.



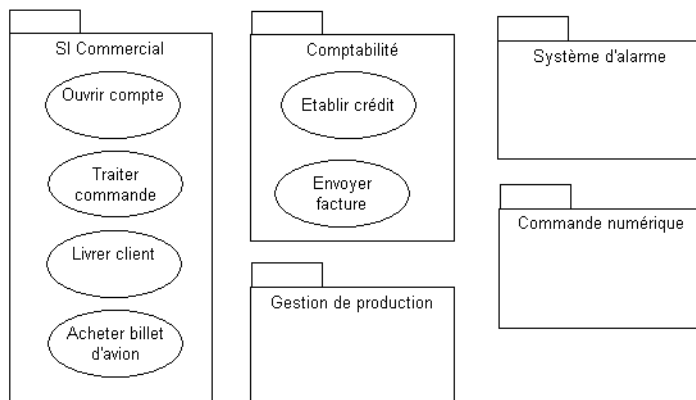
☺ Ce zoom sur la description d'un objet est disponible sur tous les éléments décrits par un diagramme.


Les paquetages



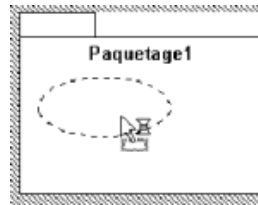
Un paquetage partitionne le domaine d'étude et les travaux associés. Il permet de regrouper divers éléments, en particulier des cas d'utilisations et des classes. Un paquetage peut aussi contenir d'autres paquetages. Les paquetages sont liés entre eux à travers des rapports contractuels définissant leur interface.

Exemples de **paquetage** : Le système d'information commercial, la comptabilité, la gestion de production, la commande numérique d'une machine, un système d'alarme, etc.



Vous pouvez créer un paquetage à l'aide du bouton  de la barre d'outils. Vous pouvez l'agrandir pour pouvoir poser des cas d'utilisation dessus.

Vous pouvez relier un cas d'utilisation à un paquetage simplement en le posant dessus. Lorsque vous déplacez l'objet sur le paquetage, la forme du paquetage est mise en relief pour indiquer que l'objet va bien lui être relié.



☛ Si les objets reliés disparaissent sous le paquetage, cliquez sur le paquetage puis cliquez sur le bouton **Arrière-plan** de la barre de dessin.

Lorsque vous déplacez un cas d'utilisation d'un paquetage à l'autre avec la souris, il est délié du premier et relié au deuxième. En revanche, si vous le déplacez avec les flèches du clavier, les liens ne sont pas modifiés.

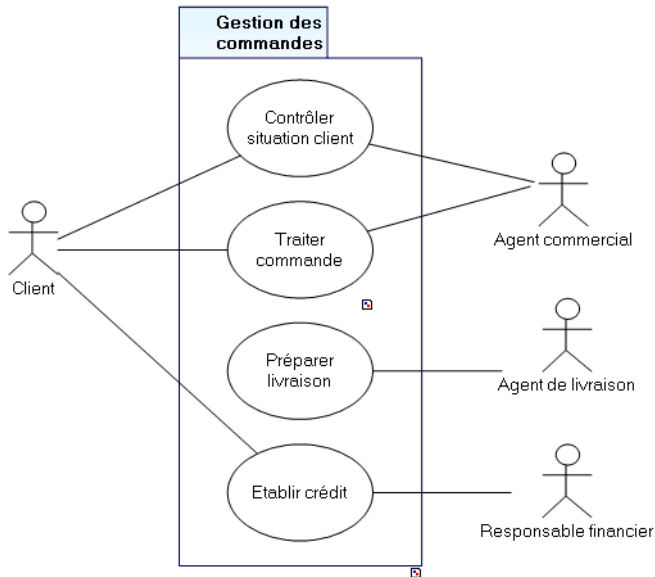
Les participations

Vous pouvez indiquer quel acteur participe à chacun des différents cas d'utilisation.



Une participation indique qu'un acteur joue un rôle dans un cas d'utilisation.


Exemple de participation




- L'agent commercial intervient dans le traitement d'une commande et dans le contrôle de la situation du client ;
- L'agent de livraison intervient dans la livraison ;
- Le responsable financier intervient dans l'établissement des crédits, etc.

Créer une participation

Pour créer une *participation*:


1. Cliquez sur le bouton **Participation**  de la barre d'objets.
2. Cliquez sur l'acteur concerné, et faites glisser la souris jusqu'au cas d'utilisation, avant de relâcher votre pression. Une fenêtre apparaît.
3. Indiquez le nom de la participation et indiquez si l'acteur en est l'initiateur.

 Il est possible de préciser le début du cas d'utilisation en activant la case à cocher **Initiateur** dans la fenêtre de propriétés de la participation correspondante. Une flèche apparaît sur le trait représentant la participation.

4. Cliquez sur **OK**.

Le lien représentant la participation apparaît dans le diagramme.

🔗 Une participation est représentée par un lien mais il s'agit d'un objet, avec des propriétés qui lui sont propres.

🔗 La bobine  n'est pas utilisée pour créer des participations. Elle sert à créer certains types de liens comme ceux entre Paquetage et les autres objets.

🔗 En cas d'erreur, vous pouvez supprimer un objet en cliquant avec le bouton droit sur cet objet, et en sélectionnant la commande **Supprimer** dans le menu contextuel. Vous pouvez de même supprimer un lien en cliquant avec le bouton droit sur le lien, et en sélectionnant la commande **Supprimer** ou **Délier** dans le menu contextuel du lien.

Multiplicités d'une participation

Sur une participation, il est possible de préciser la multiplicité :

- De l'acteur, afin d'indiquer que plusieurs instance de l'acteur sont impliquées dans une même instance du cas d'utilisation (exemple : les participants à une réunion).
- Du cas d'utilisation, afin d'indiquer qu'une même instance d'acteur intervient dans plusieurs instances du cas d'utilisation (exemple : un agent commercial qui traite plusieurs commandes d'un même client).

Pour définir les multiplicités sur une participation :

- Faites un clic droit sur la participation et sélectionnez **Propriétés**.
- Cliquez sur l'onglet **Caractéristiques**.
Un premier cadre vous permet de définir la multiplicité de l'acteur, un second cadre permet de définir celle du cas d'utilisation.

Une fois définie, la multiplicité apparaît dans le diagramme.

Les associations entre cas d'utilisation : extension et inclusion

Lorsque la taille du système à décrire est importante, il est utile d'avoir des mécanismes de représentation adaptables au niveau de détail désiré. C'est ce que permettent les associations entre cas d'*utilisation*.

Lorsqu'un cas d'utilisation comprend trop de possibilités et d'exceptions, ces dernières sont représentées à part dans des extensions du cas d'utilisation standard.

Relation d'inclusion

Un cas d'utilisation peut être sollicité automatiquement à la suite d'un autre, par exemple, la validation d'une commande inclut obligatoirement la sélection d'un moyen de paiement.

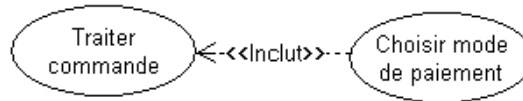
Pour indiquer qu'un cas d'utilisation en inclut un autre :

1. Dans le diagramme de cas d'utilisation, cliquez sur le bouton **Cas inclus**



2. Cliquez dans le cas utilisateur, par exemple "Traiter commande" et faites glisser la souris jusqu'au cas utilisé, par exemple, "Choisir mode de paiement", avant de relâcher votre pression.

Le lien apparaît alors dans le dessin accompagné du mot "Inclut".

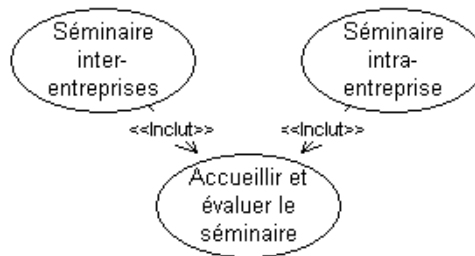


Exemples d'inclusion :

- Dans un organisme de formation, les cas d'utilisation :
 - Séminaire inter-entreprises (dont les participants viennent de plusieurs entreprises différentes)
 - Séminaire intra-entreprise (dont les participants viennent tous de la même entreprise)

peuvent avoir en commun le cas d'utilisation :

- Accueillir et évaluer le séminaire



- Dans une entreprise commerciale, le cas d'utilisation :
 - Passer une commande

peut réutiliser les cas d'utilisation suivants :

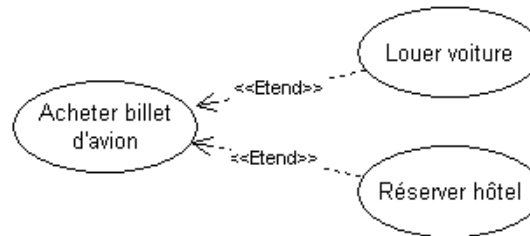
- Fournir des informations sur le client
- Passer un ordre de production
- Proposer un mode de paiement

Relation d'extension


Un cas d'utilisation peut entraîner l'exécution d'un autre. Contrairement à l'inclusion qui est automatique, l'extension est optionnelle.

Exemple d'extension :

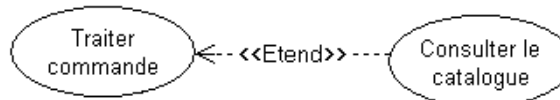
- L'achat d'un billet d'avion peut être complété par la réservation d'un hôtel ou la location d'une voiture.



Pour indiquer qu'un cas d'utilisation est l'extension d'un autre :

1. Cliquez sur le bouton **Extension** 
2. Cliquez dans un cas d'utilisation, par exemple "Consulter le catalogue" et faites glisser la souris jusqu'au cas étendu, par exemple, "Traiter commande" avant de relâcher votre pression. La fenêtre de création d'une extension apparaît. Vous pouvez définir une contrainte ou un point d'extension.
3. Cliquez sur **Terminer**.

Le lien apparaît dans le diagramme accompagné du mot "Etend".



Point d'extension

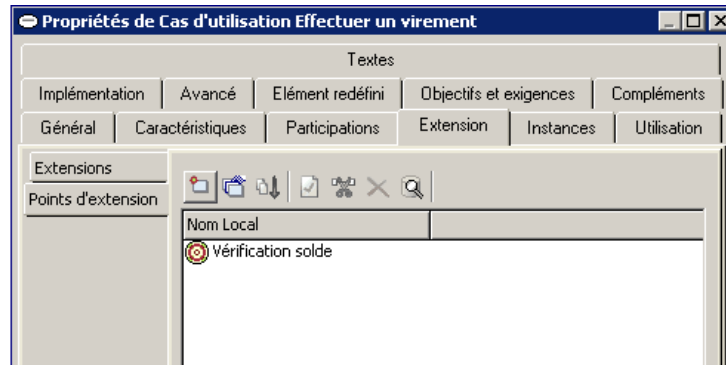
L'extension peut intervenir à un point précis du cas étendu. Ce point est appelé point d'extension.

Pour créer un point d'extension sur le cas étendu :

1. Ouvrez la fenêtre de propriétés du cas d'utilisation étendu.
2. Cliquez sur l'onglet **Extension**, et le sous-onglet **Points d'extension**.

3. Cliquez sur le bouton **Nouveau**.

Le point d'extension apparaît dans la fenêtre de propriétés. Vous pouvez le renommer.

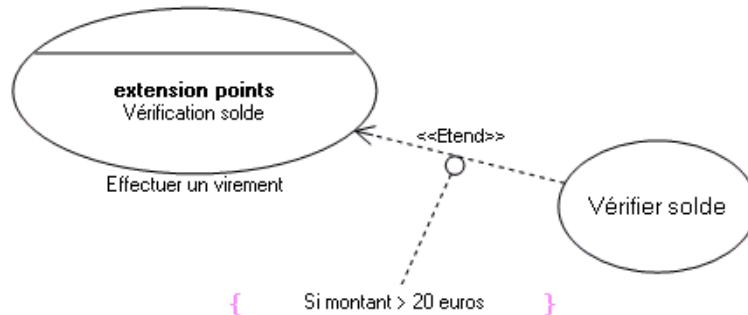


Un point d'extension peut être associé à une *contrainte* qui indique le moment où l'extension intervient. Vous pouvez ajouter une contrainte lors de la création de l'extension, ou ultérieurement, dans la fenêtre de propriétés du lien d'extension.



Une contrainte représente un contrôle ou une règle de gestion qui doit être appliquée lors de l'exécution d'un traitement.

L'exemple suivant présente le cas d'utilisation d'un virement bancaire ; au delà de la somme de 20 euros, la vérification de solvabilité du client est déclenchée.



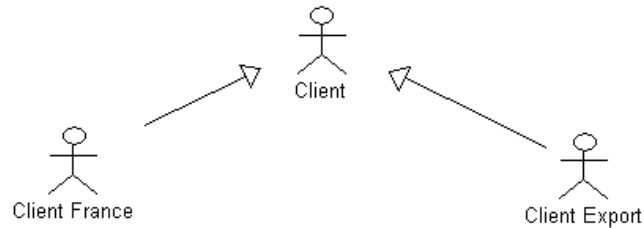
Les généralisations



Une généralisation représente une relation d'héritage entre une classe générale et une classe plus spécifique. La classe spécifique est cohérente avec la classe plus générale et en hérite ses caractéristiques et son comportement. Elle comporte cependant des informations supplémentaires. Toute instance de la classe spécifique est aussi une instance de la classe générale.


La notion de *généralisation*, initialement utilisée pour les classes, a été étendue à d'autres concepts d'UML comme acteur et cas d'utilisation.

Exemples de généralisation entre acteurs :

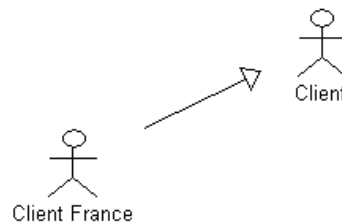


L'acteur "Client" peut être spécialisé en France et à l'Export.

Pour créer une généralisation entre acteurs :

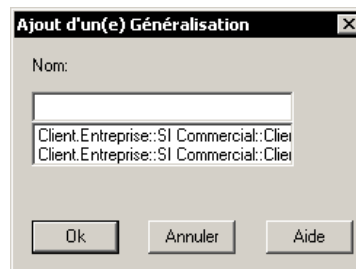
- Cliquez sur le bouton  et tirez le lien en partant de l'acteur particulier (Ex : Client France) vers l'acteur général (Ex : Client).

La généralisation apparaît dans le dessin.



☛ Vous pouvez créer de la même manière une généralisation entre deux cas d'utilisation.


Lors de la création d'une deuxième généralisation, une fenêtre vous propose de réutiliser la généralisation existante si elle porte sur le même sujet.



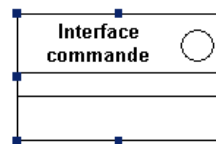
Les interfaces

Il est possible de compléter la description d'un cas d'utilisation ou d'un acteur par la description des *interfaces* par lesquelles il communique avec son environnement.


Pour créer une interface :

1. Cliquez sur le bouton **Interface**  dans la barre d'objets.
*Si le bouton **Interface** n'apparaît pas dans la barre d'objets, cliquez sur **Affichage > Vues et détails** et dans la fenêtre qui s'affiche, cochez la case **Classes**.*
2. Cliquez sur le plan de travail du diagramme.
3. Dans la fenêtre qui s'ouvre, saisissez le nom de l'interface et cliquez sur le bouton **Créer**.

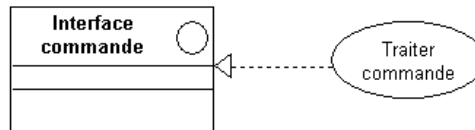
L'interface apparaît sur le diagramme.



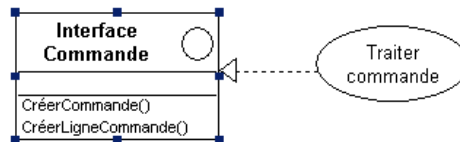
Pour préciser l'interface **supportée** par un cas d'utilisation :

- Cliquez sur le bouton  et tirez le lien en partant du cas d'utilisation (Ex : Traiter Commande) vers l'interface (Ex : Interface Commande).


Le lien apparaît dans le dessin.



Vous pouvez préciser quelles sont les opérations que ce cas d'utilisation peut réaliser par l'intermédiaire de cette interface.



Pour indiquer l'interface **requise** par un cas d'utilisation :

- Cliquez sur le bouton  et tirez le lien entre le cas d'utilisation et l'interface.
- ✓ Pour plus de détails sur les interfaces requises et supportées, voir aussi ["Relier les interfaces aux autres objets", page 102.](#)

LE DIAGRAMME DE CLASSES



Un diagramme de classes permet de représenter la structure statique d'un système, en particulier les types d'objets manipulés dans le système, leur structure interne et leurs relations. Un diagramme d'objets montre des exemples illustrant ce diagramme de classes.

La spécification des diagrammes de classes est souvent considérée comme la partie la plus importante dans la modélisation d'un système d'information. Les points abordés dans ce chapitre sont :

- ✓ ["Présentation du diagramme de classes", page 36](#)
- ✓ ["Créer un diagramme de classes", page 38](#)
- ✓ ["Les classes", page 39](#)
- ✓ ["Les attributs", page 46](#)
- ✓ ["Les opérations", page 51](#)
- ✓ ["Les signaux", page 58](#)
- ✓ ["Les associations", page 61](#)
- ✓ ["Les généralisations", page 74](#)
- ✓ ["Spécifier les interfaces", page 79](#)
- ✓ ["Spécifier les dépendances", page 81](#)
- ✓ ["Spécifier des classes paramétrées", page 82](#)
- ✓ ["Les contraintes", page 84](#)
- ✓ ["Le diagramme d'objets", page 85](#)
- ✓ ["L'éditeur de classes", page 90](#)
- ✓ ["Générer un diagramme de classes", page 94](#)

PRÉSENTATION DU DIAGRAMME DE CLASSES

Un diagramme de classes permet de représenter la structure statique d'un système, en particulier les types d'*objets* manipulés dans le système, leur structure interne et leurs relations.



Un objet est une entité avec une identité et des frontières clairement définies dont l'état et le comportement sont encapsulés. Son état est défini par les valeurs de ses attributs et de ses liens avec d'autres objets. Son comportement est représenté par ses opérations et ses méthodes. Un objet est une instance de classe.

Exemples d'objets :

- Objets de gestion :
 - Jacques Dupond, Pierre Durand, Paul Smith sont des instances de la classe "personne".
 - Les commandes no 10533 et 7322 sont des instances de la classe "commande".
 - Ecran SPD-1730 est une instance de la classe "article".
- Objets techniques utilisés pour la programmation :
 - Dlg_Order_Create, Dlg_Customer_Query sont des instances de la classe "fenêtre".
 - Str_Customer_Name, Str_Product_Comment sont des instances de la classe "chaîne".

Modéliser les données consiste à identifier les classes considérées d'intérêt pour représenter l'activité de l'entreprise, et à définir les associations qui existent entre elles.

Il faut que les classes et les associations qui constituent le diagramme de classes associé à un secteur de l'entreprise suffisent à le décrire complètement sur le plan sémantique.

En d'autres termes, on doit pouvoir décrire l'activité de l'entreprise en utilisant seulement ces classes et associations.

Ceci n'implique pas que, pour chaque mot ou verbe utilisé pour cette explication, il y ait un correspondant direct dans le diagramme de classes. Il s'agit de pouvoir traduire ce que l'on veut exprimer, au travers des classes et des associations.

La spécification des diagrammes de classes est souvent considérée comme la partie la plus importante dans la modélisation d'un système d'information.

Un diagramme d'objets montre des exemples illustrant ce diagramme de classes.

En particulier, il est possible d'illustrer un diagramme de classes par le diagramme d'objets correspondant dans un même dessin.

Le diagramme de classes : synthèse

Dans **MEGA for UML**, un diagramme de classes inclut :

- Des classes, qui représentent les concepts de base (client, compte, produit, etc.).
- Des associations, qui définissent les relations entre les différentes classes.
- Des attributs, qui décrivent les caractéristiques des classes et, dans certains cas, des associations.
- Des opérations, qui peuvent être effectuées sur les objets de la classe.

Le diagramme de classes est complété avec la définition des multiplicités.

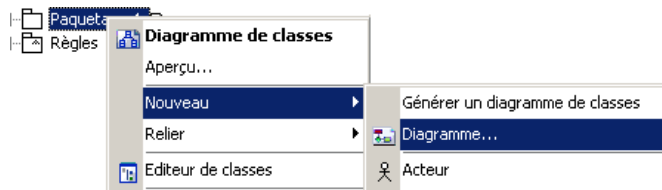
A la fin de ce guide, un glossaire rappelle la définition de chacun des concepts utilisés.

CRÉER UN DIAGRAMME DE CLASSES

Un diagramme de classe se crée depuis un paquetage.

Pour créer un diagramme de classes :

1. Cliquez avec le bouton droit sur le nom du paquetage.
2. Dans le menu contextuel qui s'affiche, cliquez sur **Nouveau** > **Diagramme...**



☛ Le menu contextuel peut être ouvert en cliquant sur un paquetage dans le navigateur ou dans un diagramme qui contient le paquetage.

3. Dans la fenêtre qui apparaît, sélectionnez "Diagramme de classes" et cliquez sur le bouton **Créer**.

Le nouveau diagramme de classes s'ouvre.

Un diagramme de classes peut décrire un paquetage, un cas d'utilisation, une classe ou une instance.

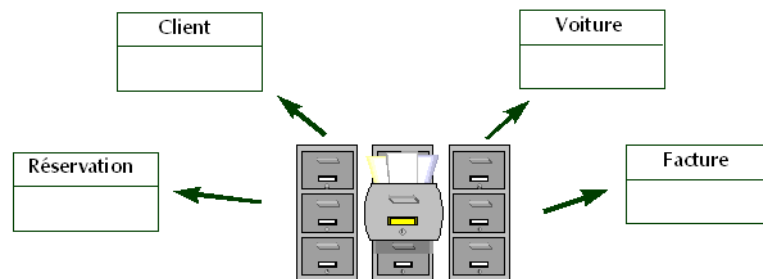
LES CLASSES

Définition d'une classe

Une *classe* est décrite par une liste d'attributs et d'opérations.

Une classe est reliée à d'autres classes via des *associations*. L'ensemble des classes et des associations constitue le noyau d'un diagramme de classes.

Nous pouvons illustrer la notion de classe en comparant les classes à des fiches dans des tiroirs.



Les classes peuvent être des objets techniques utilisés pour la programmation.

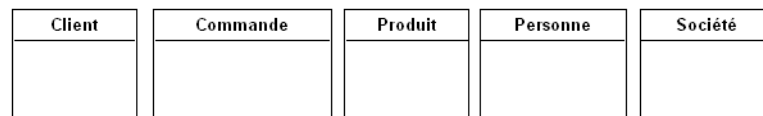
Exemples : fenêtre, rectangle, chaîne, tableau, etc.

Les classes peuvent représenter des objets techniques utilisés dans l'industrie.

Exemples : Alarme, Capteur, Zone

Les classes peuvent également représenter des objets de gestion.

Exemples : client, commande, produit, personne, société, etc.




Une classe peut aussi être la traduction d'un processus comme "Valider demande client" ou l'implémentation d'une règle de gestion fonctionnelle comme "Cohérence comptes analytiques".


☛ A la fin de ce guide, un glossaire rappelle la définition de chacun des concepts utilisés.


Créer une classe

Pour créer une classe :

1. Cliquez sur le bouton **Classe**  de la barre d'insertion d'objets.
2. Puis cliquez sur le plan de travail du diagramme.
La fenêtre **Ajout d'une classe** s'ouvre.
3. Saisissez le **Nom** de la classe et cliquez sur le bouton **Créer**.
La classe est alors posée dans le diagramme.

*Dans les exemples présentés dans ce guide, les noms des objets incluent, entre autres, des blancs, des majuscules et des caractères accentués. Il est important de noter que, lorsqu'il y a, pour un outil de génération qui utilise les spécifications effectuées avec **MEGA for UML**, des restrictions sur les caractères autorisés et sur la longueur des noms, il est préférable de se conformer à ces règles dès la spécification avec **MEGA for UML**.*

Vous pouvez créer plusieurs classes à la suite sans revenir à la barre d'outils en double-cliquant sur le bouton .

Pour revenir ensuite au mode normal, cliquez sur la flèche .

Vous pouvez utiliser partout le nom complet d'une classe en indiquant le paquetage auquel elle appartient séparé par deux fois deux points.

Exemple :

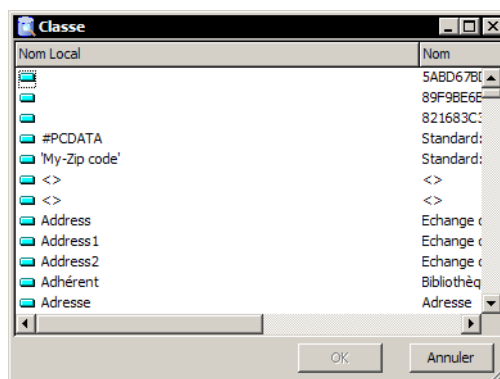
Entreprise::Gestion Commerciale::Client

Si l'un des paquetages cités n'existe pas, il est automatiquement créé et relié à la classe.

Retrouver une classe existante

Pour retrouver une classe existante:

1. Dans la fenêtre **Ajout d'une classe**, sélectionnez la commande **Lister** à l'aide de la flèche.
La liste des classes apparaît.



2. Sélectionnez la classe qui vous intéresse et cliquez sur **OK**.
Le nom de la classe sélectionnée apparaît dans la fenêtre d'ajout d'une classe.
3. Cliquez sur **Relier**.
La classe apparaît dans le diagramme.

☛ Notez que vous pouvez faire apparaître la liste des classes existantes en utilisant la touche <Ctrl-L>.

Si la base contient de nombreuses classes, vous pouvez utiliser la fonction de recherche à l'aide de la touche <Ctrl-Q> pour effectuer une sélection plus fine. Voir le guide **MEGA Publisher** pour l'utilisation de cette fonctionnalité.

Propriétés d'une classe

Les propriétés affichées varient en fonction du stéréotype de la classe.

Pour ouvrir la fenêtre de propriétés d'une classe :

- Cliquez avec le bouton droit sur la classe et sélectionnez **Propriétés**.
Plusieurs onglets permettent de définir les propriétés d'une classe.

Propriétés de Classe Exemple

Textes

Général | Références | Port | Élément redéfini | Objectifs et exigences | Compléments

Général | Caractéristiques | Attributs | Classes associées | Opérations | Instances

Nom Local: Exemple

Définition: []

Stéréotype: []

Identification de classe: Implicite

☐ Racine ☐ Feuille

☐ Abstraite ☐ Active

☒ Persistante

Paramètres: []

Classe
Une classe est un regroupement d'objets possédant des caractéristiques communes et un

OK Annuler Appliquer Aide

☛ Le commentaire situé au bas de la fenêtre décrit le champ que vous avez sélectionné.

Onglet Caractéristiques

L'onglet **Caractéristiques** permet de saisir différentes caractéristiques de la classe :

- Son **Nom**, que vous pouvez modifier.



Vous pouvez également modifier le nom d'une classe en cliquant directement sur son nom dans le dessin.

- Le **paquetage** auquel appartient la classe.
- La **Visibilité** de la classe par rapport au paquetage :
 - "Publique" : la classe est visible par tout élément situé à l'extérieur du paquetage. C'est la visibilité sélectionnée par défaut.
 - "Protégée" : la classe est visible par les éléments héritiers ou importateurs et amis.
 - "Privée" : la classe est seulement visible par les éléments importateurs et amis.
 - "Non spécifiée".



Les amis d'une classe sont des classes autorisées à connaître les internes de cette classe. Il est possible de préciser les amis d'une classe dans l'onglet Compléments de la fenêtre de propriétés de la classe.

- Son **stéréotype** : voir "[Stéréotype d'une classe](#)", page 43.

Les autres caractéristiques que vous pouvez préciser sont l'abstraction, la persistance et l'activité :

- Une classe **Abstraite** n'a pas d'instance. Elle n'est utilisée que pour mettre en commun des opérations ou des attributs communs à ses sous-classes.
- La **Persistance** précise si cette classe doit être conservée dans le temps ou si elle ne vit que le temps du traitement en mémoire dans l'ordinateur.
- Les instances d'une classe **Active** sont capables de déclencher des flux de contrôle sans qu'il y ait d'intervention de l'utilisateur.

Ex : Une instance de la classe imprimante peut envoyer un message "Plus de papier disponible" sur l'écran de l'administrateur du réseau.

- Une classe **Racine** est une classe ne possédant pas de sur-classe dans l'arborescence des généralisations de classe.
- Une classe **Feuille** est une classe ne possédant pas de sous-classe dans l'arborescence des généralisations de classe.

Vous pouvez également préciser les Paramètres d'une classe paramétrée (pour C++).



Voir "[Spécifier une classe paramétrée](#)", page 82 pour plus de détails.

Onglet Général

Certaines propriétés générales apparaissent sur tous les éléments du diagramme.

- Le nom du créateur de l'élément.
- Le nom du dernier modificateur de l'élément.
- Le statut modifiable ou non de l'élément qui indique si vous avez le droit de modifier les caractéristiques de cet élément lorsqu'une gestion des autorisations est mise en place.

Onglet Textes

Le texte **Commentaire** vous permet de commenter une classe.

☺ *Les commentaires permettent de compléter le diagramme lorsque des détails utiles n'apparaissent pas dans le dessin. Ces commentaires sont repris dans le document qui décrit le diagramme de classes.*

Autres onglets

D'autres onglets vous permettent de définir ou de visualiser :

- Les attributs d'une classe (voir "[Les attributs](#)", page 46)
- Les opérations d'une classe (voir "[Les opérations](#)", page 51)
- Les classes associées (voir "[Les associations](#)", page 61)
- Les instances d'une classe (voir "[Le diagramme d'objets](#)", page 85)

Stéréotype d'une classe

Un stéréotype est un type d'élément de modélisation qui permet d'étendre la sémantique du métamodèle. Les stéréotypes doivent être basés sur des types ou des classes existantes dont ils reprennent la structure. D'autres stéréotypes peuvent être créés par l'utilisateur.

Les stéréotypes disponibles pour une classe sont :

- **Acteur** : représente le rôle joué par quelque chose ou quelqu'un se trouvant dans l'environnement du système étudié.
- **Auxiliaire** : classe qui supporte une autre classe centrale ou fondamentale, généralement en implémentant un flux logique ou de contrôle secondaire.
- **Classe d'implémentation** : permet de caractériser les classes qui sont nécessaires à l'implémentation physique du système.
- **Classe Meta** : classe dont les instances sont des classes. En règle générale, les méta-classes sont utilisées pour construire des méta-modèles.
- **Contrôle** : permet de caractériser les classes effectuant des traitements internes au système. Ceux-ci nécessitent généralement le concours de plusieurs classes.
- **Entité** : permet de caractériser des classes passives qui ne génèrent aucune interaction par elles-mêmes. Elles peuvent participer à plusieurs cas d'utilisation et survivent généralement à une interaction

unique. Elles représentent des objets partagés entre les différents acteurs qui les manipulent.

- **Énumération** : type de données contenant une liste de valeurs tabulées.
- **Expression** : expressions de types de données complexes basés sur des types.
- **Focus** : classe qui définit le flux logique ou de contrôle principal pour la ou les classes auxiliaires qui la supportent.
- **Frontière** : permet de caractériser les classes qui sont en prise directe avec l'environnement du système. Les interfaces hommes-machines en font partie.
- **Interface** : une interface est constituée d'un ensemble d'opérations qui décrivent le comportement d'un élément. En particulier, une interface représente la partie visible d'une classe ou d'un paquetage dans une relation contractuelle de type client - fournisseur.

☞ *Ce sont des interfaces entre les différents composants du système informatique. Ce ne sont pas des interfaces avec les utilisateurs du système qui sont du stéréotype frontière. Voir "[Spécifier les interfaces](#)", page 79 pour plus de détails.*

- **Opérateur** : représente un acteur humain qui interagit avec le système. Un opérateur interagit avec d'autres opérateurs et manipule des entités lorsqu'il participe à la réalisation d'un cas d'utilisation.
- **Opérateur externe** : un opérateur externe interagit directement avec des acteurs extérieurs au système.
- **Opérateur interne** : un opérateur interne interagit avec d'autres opérateurs et entités à l'intérieur du système.
- **PowerType** : méta-type dont les instances sont des sous-types d'un autre type.
- **Structure** : classe servant à décrire une structure utilisée dans les programmes.
- **Thread** : stéréotype utilisé dans l'implémentation d'un objet actif en tant que processus léger.
- **Type élémentaire** : permet de caractériser les types de données.
- **Utilitaire** : une classe de ce stéréotype regroupe des variables globales et des procédures utiles à la programmation décrites sous forme des attributs et opérations de cette classe.
- **Schema group** : classe décrivant un type d'élément XML, dont les sous-éléments forment un groupe.
- **XML Document Definition Root** : classe servant à décrire la structure d'un message échangé entre deux systèmes en utilisant la syntaxe du langage XML.

Option d'affichage des stéréotypes

Une option vous permet d'afficher les stéréotypes dans la fenêtre de navigation des objets.



Pour activer cette option :

1. Depuis l'espace de travail **MEGA**, cliquez sur le menu **Outils** > **Options**.
2. Dans la partie gauche de la fenêtre des options, sélectionnez le dossier **Espace de travail**.
3. Dans la partie droite, cochez l'option **Afficher le stéréotype des objets UML dans le navigateur**.
4. Cliquez sur **OK**.

LES ATTRIBUTS

Définition d'un attribut

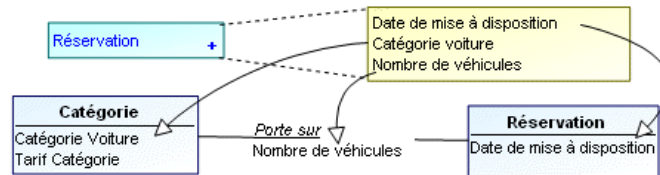
Un attribut est une propriété nommée d'une classe. C'est la donnée élémentaire mémorisée dans le système d'information de l'entreprise.

Exemples :

"Nom du client" (Attribut de la classe client).

"No client" (Identifiant de la classe client).

"Solde du compte" (Attribut de la classe compte).

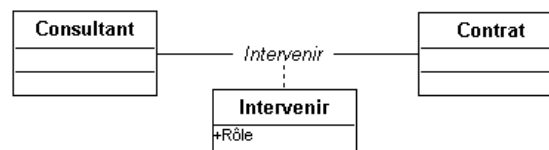


Les classes et les classes d'associations peuvent être caractérisées par des attributs.

Ces attributs ont pu, par exemple, être révélés par l'étude du contenu des messages qui circulent à l'intérieur de l'entreprise.

Un attribut est porté par une association quand sa valeur dépend de l'ensemble des classes participant à cette association.

Dans le diagramme présenté ci-après, le "rôle" qu'un "Consultant" a joué sur un "Contrat" dépend du consultant et du contrat, donc de l'association "Intervenir".



Spécifier les attributs d'une classe

Pour définir les attributs d'une classe :

1. Cliquez sur la classe avec le bouton droit de la souris.
2. Dans le menu contextuel, sélectionnez **Attributs**.

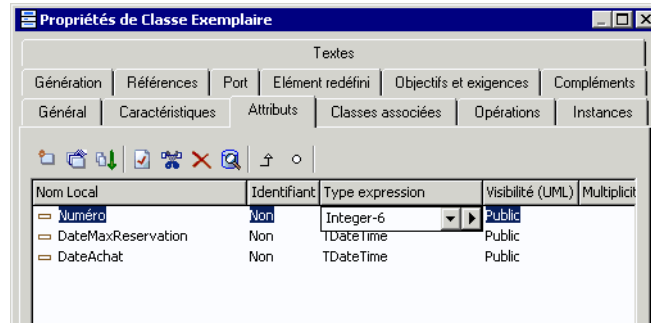
Les attributs de la classe y sont présentés. Il est possible de préciser pour chacun d'eux :

- Son **Type**, qu'il est possible d'exprimer sous forme d'une expression.

Ex : Integer.

☛ L'expression doit être conforme à la syntaxe UML. Voir "[Signature d'une opération ou d'un signal](#)", page 53 pour plus de détails.

Pour consulter la liste des types possibles, cliquez dans le champ correspondant et déplacez-vous à l'aide de la flèche du menu déroulant.



Sont présents dans la liste les types qui appartiennent aux différentes classes du paquetage de départ (il s'agit ici de "Bibliothèque").

Pour rechercher d'autres types, cliquez sur la flèche  à droite du menu déroulant puis sélectionnez **Rechercher**. La fenêtre **Rechercher** s'affiche.

Vous pouvez également taper directement dans le champ le nom du type de l'attribut.

- Sa **Visibilité** :
 - "Public": c'est la visibilité par défaut. L'attribut est visible par tous.
 - "Protégé": l'attribut est visible par les héritiers de son paquetage ou ses amis.
 - "Privé": l'attribut est visible par sa classe ou ses *amis*.

📖 Les amis d'une classe sont des classes autorisées à connaître les internes de cette classe. Il est possible de préciser les amis d'une classe dans l'onglet Compléments de la fenêtre de propriétés de la classe.

- Sa **Multiplicité**, c'est à dire le nombre de répétitions de cet attribut dans la classe.

Les boutons de la barre d'outils vous permettent d'effectuer les manipulations suivantes :



Créer un nouvel attribut ou plus généralement un nouvel élément.



Réordonner les éléments.



Afficher les propriétés d'un élément.



Supprimer un élément.



Explorer un élément.

Attributs hérités

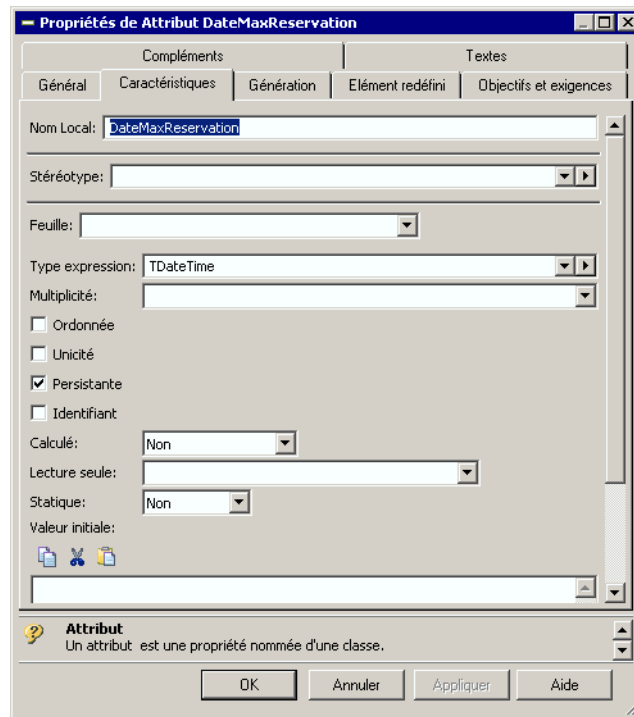
Lorsqu'une généralisation existe entre une classe générale et une classe particulière, la classe particulière hérite des attributs de la classe générale.

- Cliquez sur le bouton **Attributs hérités**  pour visualiser les attributs hérités d'autres classes.

Propriétés des attributs

Pour ouvrir la fenêtre de propriétés d'un attribut :

- Cliquez avec le bouton droit sur l'attribut et sélectionnez **Propriétés**.



Dans cette fenêtre, vous pouvez préciser :

- Le **Type** de l'attribut sous la forme d'une **expression** (voir "Types", page 50).
- S'il s'agit d'un attribut **statique** : indique si l'attribut peut prendre des valeurs spécifiques pour chacune des instances de la classe ou bien avoir une valeur qui caractérise l'ensemble de la classe.
 - "Oui" : l'attribut a une valeur qui caractérise l'ensemble de la classe. Par exemple, l'attribut "Longueur des numéros de téléphone" de la classe "Client France" est de 10 chiffres.
 - "Non" : l'attribut peut prendre une valeur différente pour chacune des instances de la classe. Par exemple, l'attribut "Numéro de téléphone" prend une valeur différente pour chaque instance de la classe "Client".
- S'il s'agit d'un attribut **Persistant**, c'est-à-dire si la valeur de cet attribut doit être conservée dans le temps ou si elle ne vit que le temps du traitement en mémoire dans l'ordinateur.
- Sa **Multiplicité**, c'est-à-dire le nombre de répétitions de cet attribut dans la classe.
- S'il est en **Lecture seule**, c'est-à-dire si sa valeur peut être modifiée après avoir été renseignée une première fois.

- S'il s'agit d'un **Attribut Calculé**, c'est-à-dire que sa valeur est déduite de la valeur d'un ou plusieurs autres attributs.
- La **Valeur initiale** de l'attribut, qu'il prendra lors de la création d'une instance de la classe.
- Sa **Visibilité** :
 - "Public" : c'est la visibilité par défaut. L'attribut est visible par tous.
 - "Protégé" : l'attribut est visible par les héritiers de son paquetage ou ses amis.
 - "Privé" : l'attribut est visible par sa classe ou ses amis.

Types

Un type permet de mettre en commun des caractéristiques communes à plusieurs attributs. Un type est implémenté sous forme de classe.

Toute classe peut être utilisée pour typer un attribut ou un paramètre.

Exemple : Client, Commande, Fenêtre, Tableau

Les classes de stéréotype "Type élémentaire" sont créées uniquement afin de typer des attributs ou des paramètres. Elles sont invariables.



Exemples de types élémentaires :

Chaîne.


Entier.

Adresse Export.

Montant en devises.

Vous pouvez lister les types existants à l'aide de la flèche  ou en créer de nouveaux à l'aide de la flèche .

Les types proposés sont les classes détenues ou utilisées par le paquetage ou par les paquetages dont il est client.

 Pour renseigner la structure d'un type, posez la classe correspondante dans le même diagramme ou dans un autre diagramme et activez la commande Propriétés de son menu contextuel.

Des informations complémentaires sur l'utilisation des types élémentaires sont disponibles dans l'annexe de ce guide.

LES OPÉRATIONS

Définition d'une opération

Une opération est un service qui peut être demandé à un objet pour mettre en oeuvre un comportement défini. Une opération possède une signature qui permet de préciser les paramètres qui lui sont nécessaires.

Exemples :

"Calcul Age" (Opération de la classe client).

"Imprimer" (Opération de la classe dessin).

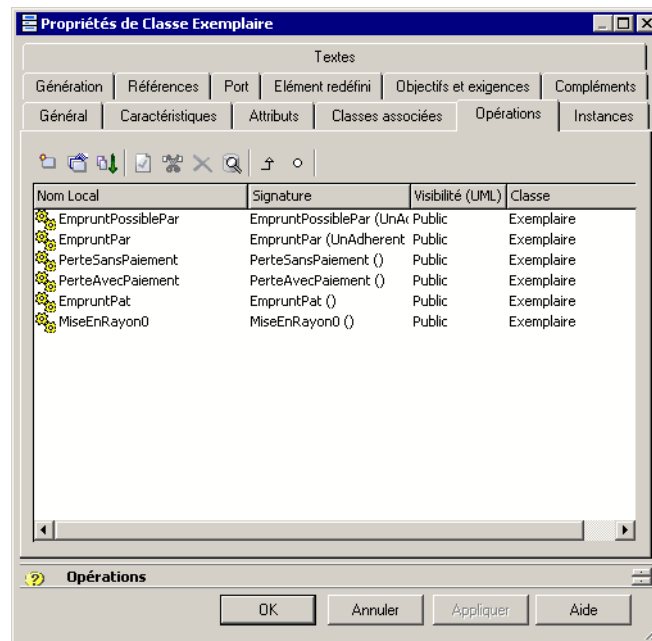
"Calculer échéancier" (Opération de la classe prêt).


Spécifier les opérations d'une classe

Pour spécifier les opérations d'une classe :

- Cliquez avec le bouton droit sur la classe et sélectionnez **Opérations**.


Une fenêtre présentant les opérations de la classe s'ouvre.



Vous pouvez ajouter des opérations en cliquant sur le bouton  et préciser leur signature.

Opérations héritées

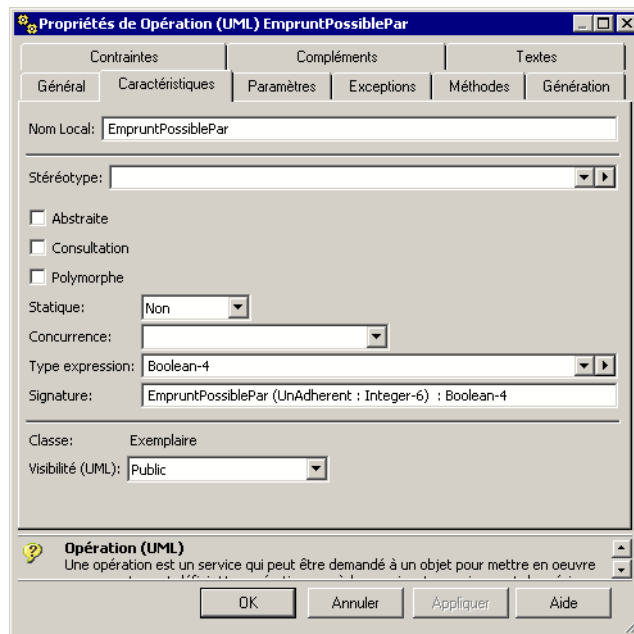
Lorsqu'une généralisation existe entre une classe générale et une classe particulière, la classe particulière hérite des opérations de la classe générale.

- Cliquez sur le bouton **Opérations héritées**  pour visualiser les opérations héritées d'autres classes.

Propriétés d'une opération

Pour ouvrir la fenêtre de propriétés d'une opération :

- Cliquez avec le bouton droit sur l'opération et sélectionnez **Propriétés**.



Vous pouvez indiquer pour chaque opération :

- Son **Stéréotype** afin de préciser son utilisation :
 - **Constructeur** : créer une instance de la classe.
 - **Destructeur** : détruire une instance de la classe.
 - **Itérateur** : parcourir les instances de la classe.
 - **Sélecteur** : effectuer une sélection parmi les instances de la classe.
- S'il s'agit d'une opération **Statique** : si l'opération qui peut prendre des valeurs spécifiques pour chacune des Instances de la classe ou bien avoir une valeur qui caractérise l'ensemble de la Classe.
- La **Concurrence** permet de préciser le comportement de l'opération lorsqu'elle est appelée plusieurs fois simultanément.
 - **Concurrente** : l'opération répond simultanément aux différents appels.

- **Protégée** : l'opération répond au premier appel et rejette les suivants.
- **Séquentielle** : l'opération répond successivement à chacun des appels.
- Si c'est une opération en **Consultation**, c'est-à-dire qui ne modifie pas l'état de l'objet.
- Si cette opération est **Polymorphe**, c'est-à-dire que des méthodes peuvent être redéfinies pour cette opération dans des sous-classes.
- Sa **Visibilité** :
 - **Publique** : c'est la visibilité par défaut. L'opération est visible par tous.
 - **Protégée** : l'opération est visible par les héritiers de son paquetage ou ses amis.
 - **Privée** : l'opération est visible par sa classe ou ses amis.

Les indications suivantes sont utilisées pour compléter la signature de l'opération.

- Le **Type expression** de l'opération.



Le type expression d'une opération précise le type de la variable retournée par l'opération à la fin de son exécution.

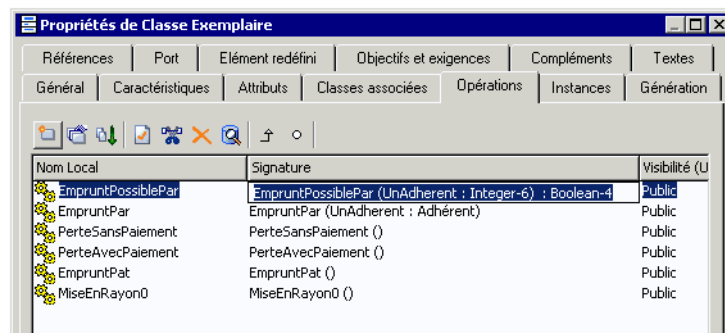
- Sa **Signature**.

Signature d'une opération ou d'un signal

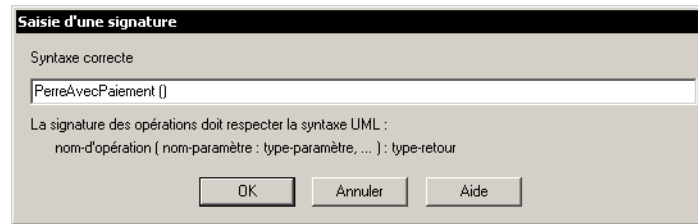
La signature d'une opération ou d'un signal est constituée du nom de l'opération (du signal), de son type de retour, et de ses paramètres avec leurs types. La syntaxe UML standard est utilisée pour cette signature, elle est du type : OpeO (Param0 : M-Bool) : M-Bool.

Il est possible de définir la signature :

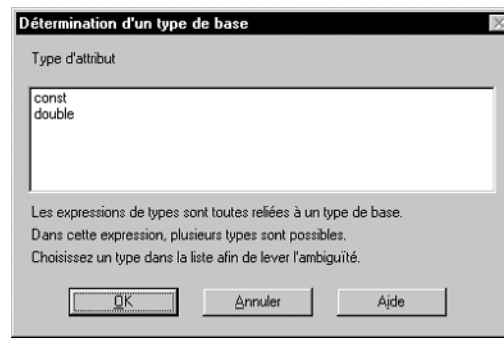
- Soit dans la fenêtre de propriétés de l'opération ou du signal.
- Soit dans la fenêtre de propriétés de la classe à laquelle appartient l'opération, en saisissant directement la **Signature** dans l'onglet **Opérations**.



Lorsque la signature est directement saisie, un contrôle de sa validité est effectué. En cas d'erreur, une fenêtre précise la nature de l'erreur et permet de la corriger.



De même, lorsque la signature comprend un type suivi ou précédé de complément (par exemple, Ope1(param1: const double), il est nécessaire de lever l'ambiguïté, en choisissant parmi un des mots composant l'expression.



Si le type n'existe pas, il est automatiquement créé.

☛ Il est possible de filtrer les mots qui ne correspondent jamais à un type en créant un objet de type `"_UML ReservedWord"` et en lui donnant comme nom le mot à filtrer (en métamodèle avancé : Outils > Options : Référentiel).

La signature qui est conservée inclut une référence au type : si le type est renommé, les signatures qui l'utilisent reflètent cette évolution.

Syntaxe des signatures

La syntaxe standard des signatures est :

```
nomopération(paramètre1:expressiondutype1,paramètre2:expres  
siondutype2,...):expressiondutyperetour
```

Les noms comportant des blancs ou des caractères spéciaux doivent être placés entre apostrophes ('Nom du client'). Lorsqu'un nom contient une apostrophe, il faut dupliquer l'apostrophe : 'Nom de l' 'acheteur'

Exemples de signatures :

```
Déstocker (Produit0 : Entier(3), Quantité0 : Entier) :  
Booléen
```

```
'Création de commande' ('Nom du client' : Client) : Byref  
Variable
```

Dans la spécification d'une signature, il est possible de préciser le paquetage auquel appartient une classe, séparé par deux fois deux points.

Exemple : `Entreprise::'Gestion Commerciale'::Client.`

La classe citée est reliée au paramètre ou au type de retour. Si elle n'existe pas, elle est créée. De même, les paquetages cités dans le parcours qui n'existent pas sont créés et reliés à la classe.

Si le paquetage n'est pas précisé, une fenêtre vous propose de choisir entre les éventuelles classes homonymes.

Paramètres d'une opération

Dans la fenêtre de propriétés d'une opération, l'onglet **Paramètres** vous permet de préciser :

- Son **Type** sous forme d'une expression Ex : `Integer(5)`.
- Sa **Valeur par défaut** Ex : `0`.
- Sa **Direction** : en entrée et/ou en sortie de l'opération.

Pour créer un nouveau *paramètre* :

- Faites un clic droit dans la fenêtre et sélectionnez **Nouveau**.



Un paramètre est la spécification d'une variable qui peut être modifiée, transmise ou retournée. Un paramètre peut préciser un nom, un type et une direction. Les paramètres sont utilisés pour les opérations, les messages et les événements.



Un argument est une valeur spécifique correspondant à un paramètre.

Méthodes d'une opération (comportement opaque)

Une méthode - ou comportement opaque - est une représentation textuelle de l'implémentation d'une opération, d'une classe ou d'un composant. Elle spécifie l'algorithme ou la procédure qui produit les résultats d'une opération ou le comportement d'un élément.

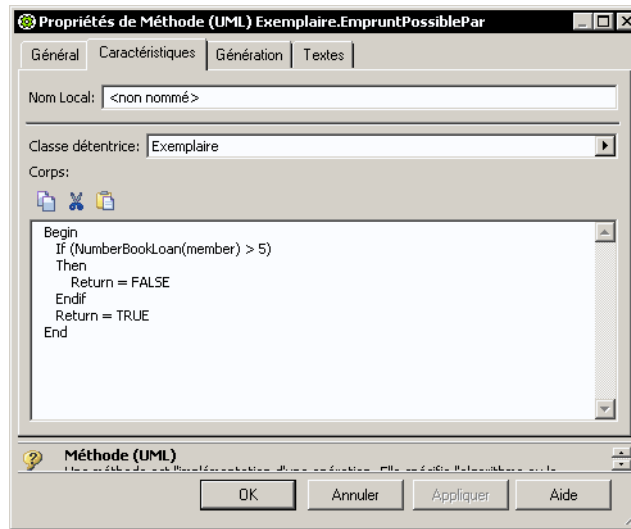
Pour définir la méthode implémentant une opération :

1. Ouvrez la fenêtre de propriétés de l'opération.
2. Sous l'onglet **Méthode**, cliquez sur le bouton **Nouveau**.
3. Cliquez dans la colonne **Classe détentrice** et sélectionnez **Classes candidates (Opérations)**.
4. L'outil recherche les classes susceptibles de fournir les méthodes d'implémentation.
Vous pouvez également créer une nouvelle classe détentrice de la méthode.

Pour saisir le corps du texte de la méthode qui implémente l'opération :

1. Ouvrez la fenêtre de propriétés de la méthode.
2. Cliquez le sous-onglet **Caractéristiques**.

3. Définissez la méthode dans le cadre **Corps**.



Quand une classe possède plusieurs sous-classes, chaque sous-classe est susceptible d'implémenter l'opération sous forme d'une méthode différente.

L'onglet **Méthode** présente la méthode relative à la classe sélectionnée.

Conditions d'une opération

Vous pouvez définir les conditions d'une opération sous la forme de contraintes.

L'onglet **Contraintes** de la fenêtre de propriétés de l'opération vous permet de spécifier :

- La **Pré-Condition**, qui doit être vérifiée avant que l'opération ne s'exécute.
- Le **Corps**, qui doit être vérifié lors de l'exécution de l'opération.
- La **Post-Condition** qui doit être vérifiée après l'exécution de l'opération.

Si une condition n'est pas respectée, une exception est générée.

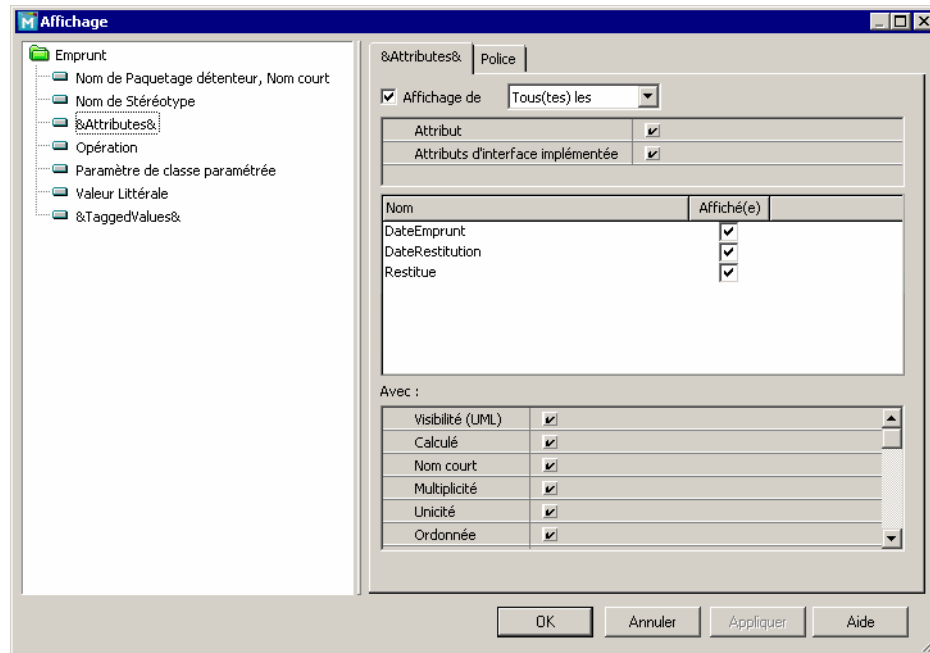
Exceptions d'une opération

L'onglet **Exceptions** vous permet de présenter les messages d'erreur envoyés par l'opération lorsqu'une exception se produit et de préciser leur signature.

Afficher les attributs et les opérations d'une classe

Pour modifier l'affichage des attributs et des opérations de la classe :

1. Cliquez avec le bouton droit sur la ou les classes dont vous voulez afficher des attributs.
2. Sélectionnez **Formes et détails**.
La fenêtre de sélection des éléments à afficher apparaît.
3. Choisissez **Attribut** dans l'arbre qui vous est présenté.



4. Sélectionnez les attributs que vous voulez afficher.

Vous pouvez les afficher **Tous**, en afficher **Certains** que vous allez sélectionner dans la liste, ou n'en afficher **Aucun**.

Vous pouvez demander l'affichage de la **Visibilité**, du **type**, ... de chacun des attributs.

Un type de données permet de mettre en commun des caractéristiques communes à plusieurs attributs. Les types de données sont implémentés sous forme de classe.

Vous pouvez cacher ou rendre visible le compartiment présentant les attributs dans le dessin de la classe, en activant ou désactivant la case à cocher devant "Affichage de"



L'affichage des opérations est indiqué de la même manière, en choisissant **Opération** dans l'arbre qui vous est présenté.

LES SIGNAUX

Définition d'un signal


Un signal est un événement qui peut être invoqué explicitement. Un signal peut posséder des paramètres. Un signal peut être envoyé à un objet ou à un ensemble d'objets. Il peut être invoqué dans le cadre de la participation d'un acteur à un cas d'utilisation.

Un signal peut être émis ou reçu par une classe. Il peut être également émis par une opération à la suite d'une exception.

Spécifier les signaux d'une classe

Créer un signal émis ou reçu

Pour préciser quels sont les signaux qui peuvent être émis ou reçus par une classe :

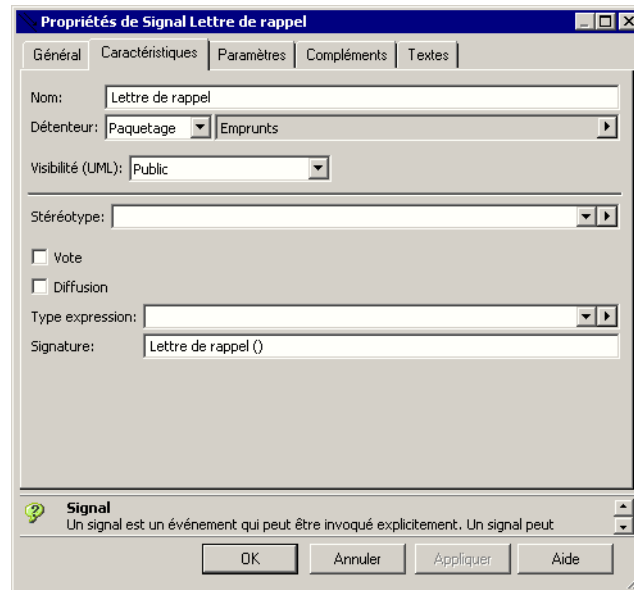
1. Ouvrez la fenêtre de propriétés d'une classe.
2. Cliquez sur l'onglet **Compléments**.
3. Dans l'arbre qui vous est présenté, sélectionnez la branche **Signal émis** ou **Signal reçu**, puis cliquez sur le bouton 
4. Indiquez le nom du signal et cliquez sur **OK**.

Propriétés d'un signal

.Pour ouvrir la fenêtre de propriétés d'un signal :

- Dans la fenêtre de propriétés d'une classe, dans l'onglet **Compléments**, cliquez sur le signal avec le bouton droit de la souris et sélectionnez **Propriétés**.

La fenêtre de propriétés du signal apparaît.



Vous pouvez indiquer pour un signal :

- Son **Stéréotype** afin de préciser son utilisation :
 - **Exception** : signal d'erreur généré lorsqu'une exception se produit pendant l'exécution d'une opération.
- Sa **Visibilité** par rapport au paquetage :
 - **Publique** : c'est la visibilité par défaut. Le signal est visible par tout élément situé à l'extérieur du paquetage.
 - **Protégée** : le signal est visible par les éléments héritiers ou amis.
 - **Privée** : le signal est visible par sa classe ou ses amis.

Les amis d'une classe sont des classes autorisées à connaître les internes de cette classe. Il est possible de préciser les amis d'une classe dans l'onglet Compléments de la fenêtre de propriétés de la classe.

- Le **Type expression** du signal (voir *type expression*).

Le type expression d'un signal précise le type de la variable retournée par le signal lors de sa réception par son destinataire.

Un signal peut être une demande de **Vote** envoyé à chacun des objets actifs pour leur demander s'il est possible d'effectuer une action, comme par exemple fermer une session Windows.

Un signal peut donner lieu à une **Diffusion** générale à tous les objets actifs.

Paramètres d'un signal

Les *Paramètres* du signal sont renseignés sous l'onglet **Paramètres** de sa fenêtre de propriétés. Vous pouvez préciser :

- Son **Type** sous forme d'une expression. Ex : Integer(5).
- Sa **Valeur** par défaut. Ex : 0.
- Sa **Direction** : en entrée et/ou en sortie de l'opération.



Un paramètre est la spécification d'une variable qui peut être modifiée, transmise ou retournée. Un paramètre peut préciser un nom, un type et une direction. Les paramètres sont utilisés pour les opérations, les messages et les événements.



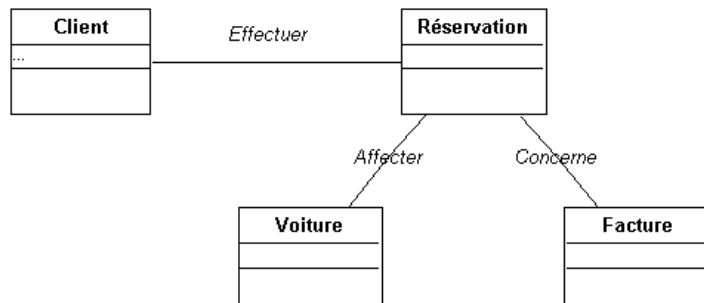
Un argument est une valeur spécifique correspondant à un paramètre.

LES ASSOCIATIONS

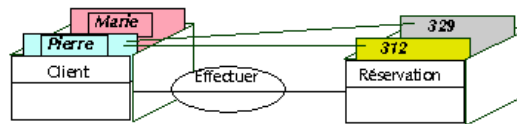
Une association est une relation existant entre deux classes.

Une association est binaire quand elle relie deux classes, ternaire quand elle en relie trois, etc.

Les associations peuvent être comparées à des liens entre des fiches.



Le dessin suivant permet de visualiser "en trois dimensions" les situations qu'un diagramme de classes permet de mémoriser.



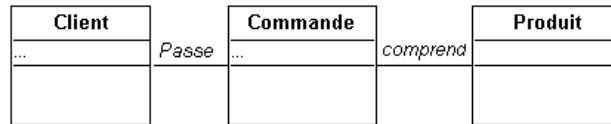
Pierre et Marie sont des clients. Pierre a effectué les réservations numéros 312 et 329.

Un diagramme de classes doit permettre de mémoriser toutes les situations du contexte de l'entreprise.

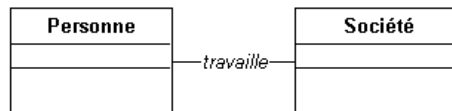
☛ Le diagramme ne doit pas permettre de représenter des situations irréalistes ou aberrantes.

Exemples d'association

- Un client *pass*e une commande.
- Une commande *comprend* plusieurs produits.



- Une personne *travaille* pour une société.




- Une alarme *est déclenchée* par un capteur.

Un capteur *couvre* une zone.

Une fenêtre *affiche* une chaîne de caractères.

Créer une association

Pour créer une *association*:

1. Cliquez sur le bouton **Association**  de la barre d'objets.
2. Cliquez dans une des classes concernées, et faites glisser la souris jusqu'à la deuxième classe avant de relâcher votre pression. L'association apparaît dans le diagramme sous forme d'un trait.

S'il existe déjà une association entre les deux classes, la fenêtre **Ajout d'une association** s'ouvre. Elle permet de créer une nouvelle association ou de choisir entre les associations existantes.

Vous pouvez préciser un nom pour l'association dans sa fenêtre de propriétés, accessible par son menu contextuel.

☛ En cas d'erreur, vous pouvez supprimer un élément ou un lien en cliquant avec le bouton droit sur cet élément, et en sélectionnant la commande *Supprimer* dans son menu contextuel.

Vous pouvez également créer une association à partir de l'éditeur de classes. Voir ["Créer des objets dans l'éditeur de classes"](#), page 92.

Les rôles des associations

Il est possible de décrire les différents *rôles* joués par les classes dans les associations, et de préciser leur multiplicité et leur navigabilité.

Chaque extrémité d'une association permet de préciser le rôle joué par chaque classe dans l'association.

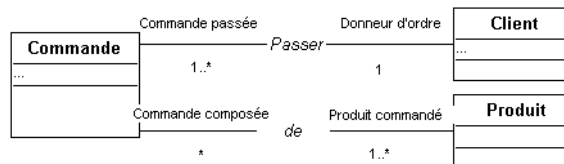
Visuellement, le nom du rôle se distingue du nom d'une association, car il est placé près de son extrémité. De plus, il apparaît en caractères droits, alors que le nom de l'association est en italique.



Lorsque deux classes sont reliées par une seule association, le nom des classes suffit souvent à caractériser le rôle ; nommer les rôles prend tout son intérêt lorsque plusieurs associations relient deux classes.

Exemples de rôles :

- Un client est le *donneur d'ordre* d'une commande.
- Une commande est *passée* par un client.
- Une commande est *composée* de produits.
- Un produit est *commandé*.



Une personne est un *employé* dans une société.

Une société est *l'employeur* de ces personnes.

Une alarme est *déclenchée* par un ou plusieurs capteurs.

Une zone est *couverte* par un capteur.

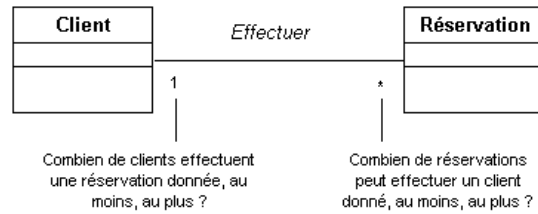
Une ou plusieurs chaînes de caractères sont *affichées* dans une fenêtre.

Multiplicité d'un rôle

📖 La multiplicité précise l'intervalle entre les valeurs minimum et maximum des cardinalités possibles pour un ensemble. On l'indique en particulier pour chacun des rôles que jouent les classes dans une association. Elle peut prendre les valeurs *, 0..1, 1, 1..*, 2..*, 4..10, etc. La valeur proposée par défaut est *.

📖 La cardinalité est le nombre d'éléments contenus dans un ensemble.

La *multiplicité* exprime le nombre minimum et maximum d'instances d'une classe pouvant être reliées par l'association à chaque instance de l'autre classe.

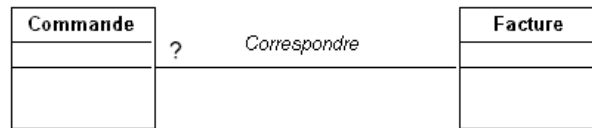


Les multiplicités usuelles sont "1", "0..1", "*" ou "0..*", "1..*", et "M..N" où "M" et "N" sont des entiers :

- La multiplicité "1" indique qu'une et une seule instance de la classe est reliée par cette association à chaque instance de l'autre classe.
- La multiplicité "0..1" indique qu'au plus une instance de la classe peut être reliée par cette association à chaque instance de l'autre classe.
- La multiplicité "*" ou "0..*" indique qu'un nombre quelconque d'instances de la classe peuvent être reliées par l'association à chaque instance de l'autre classe.
- La multiplicité "1..*" indique qu'au moins une instance de la classe est reliée par l'association à chaque instance de l'autre classe.
- La multiplicité "M..N" indique qu'au moins M instances de la classe et au plus N sont reliées par l'association à chaque instance de l'autre classe.

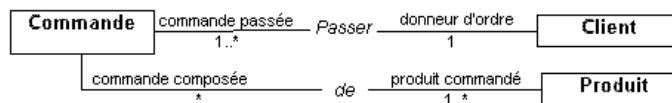
| | |
|------|-----------------------------|
| 1 | Un et un seul |
| 0..1 | Zéro ou un |
| M..N | De M à N (entiers naturels) |
| * | De zéro à plusieurs |
| 0..* | De zéro à plusieurs |
| 1..* | De un à plusieurs |

L'exemple suivant va nous permettre d'illustrer la signification de chacune des multiplicités.



- 0..1 : A une commande correspond une facture au maximum ou aucune.
- * : Aucune restriction n'est imposée sur le nombre de factures correspondant à une commande.
- 1 : A chaque commande correspond une facture et une seule.
- 1..* : A chaque commande correspond une ou plusieurs factures.

Autres exemples de multiplicité :

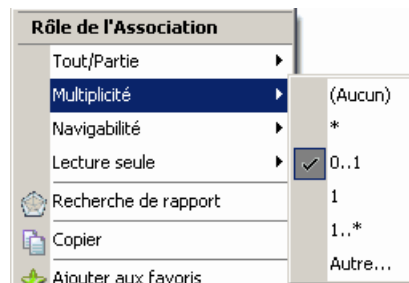


- 1..* : Un client peut passer une ou plusieurs commandes.
- 1 : Une commande est passée par un et un seul client.
- 1..* : Une commande comprend un ou plusieurs produits.
- * : Un produit peut faire partie de plusieurs commandes, ou d'aucune.
- 0..1 : Une personne travaille pour une société.
- 1..* : Une alarme est déclenchée par un ou plusieurs capteurs.
- 1 : Un capteur couvre une et une seule zone.
- 1..* : Une fenêtre affiche une ou plusieurs chaînes de caractères.

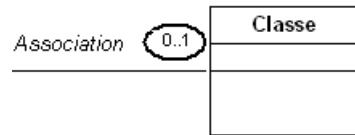
Préciser la multiplicité d'un rôle

Pour préciser la multiplicité d'un rôle :

- Cliquez avec le bouton droit sur le trait qui se trouve entre l'association et la classe, et sélectionnez **Multiplicité**.



La multiplicité apparaît alors sur le rôle.



☛ Si le menu affiché ne propose pas les multiplicités, contrôlez que vous avez bien cliqué sur le trait qui matérialise le rôle, et non sur l'association.

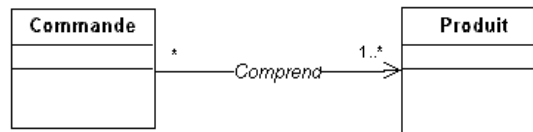
😊 Toutes les informations précisées avec le menu contextuel peuvent être consultées et modifiées dans la fenêtre de propriétés du rôle.

Navigabilité d'un rôle

La navigabilité précise le sens dans lequel l'association entre deux classes peut être parcourue. Pour ne pas encombrer le dessin, on n'indiquera la navigabilité que lorsqu'elle n'a lieu que dans un seul sens.

Exemple de navigabilité :

- Il est nécessaire de connaître tous les produits contenus dans une commande.
- Par contre, il est rarement utile de connaître toutes les commandes qui portent sur un produit.



Préciser la navigabilité d'un rôle

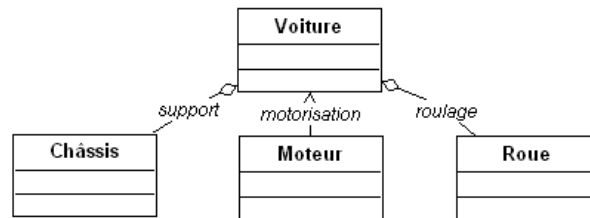
Pour indiquer qu'une association n'est navigable que dans un sens :

1. Cliquez avec le bouton droit sur le rôle non navigable.
2. Sélectionnez **Navigabilité > Non**.
Une flèche représentant la navigabilité apparaît alors sur le rôle opposé.

Agrégation d'un rôle

L'agrégation est une forme particulière d'association qui indique que l'une des classes contient l'autre.

Exemple : Une voiture comprend un châssis, un moteur et des roues.



Préciser l'agrégation d'un rôle

Pour préciser l'agrégation d'un rôle :

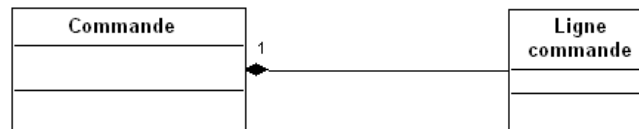
1. Cliquez avec le bouton droit sur le rôle.
2. Sélectionnez **Tout/Partie** > **Agrégat**.
Un losange représentant l'agrégation apparaît alors sur le rôle.

Composition d'un rôle

La composition est une agrégation forte pour laquelle la durée de vie des composants coïncide avec celle du composé. Une composition est une agrégation immuable avec une multiplicité 1.

Exemple : Une commande est composée de plusieurs lignes de commande qui n'existent plus si la commande est supprimée.

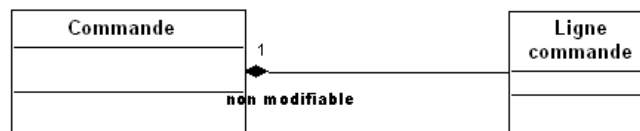
La composition est matérialisée par un losange noir.



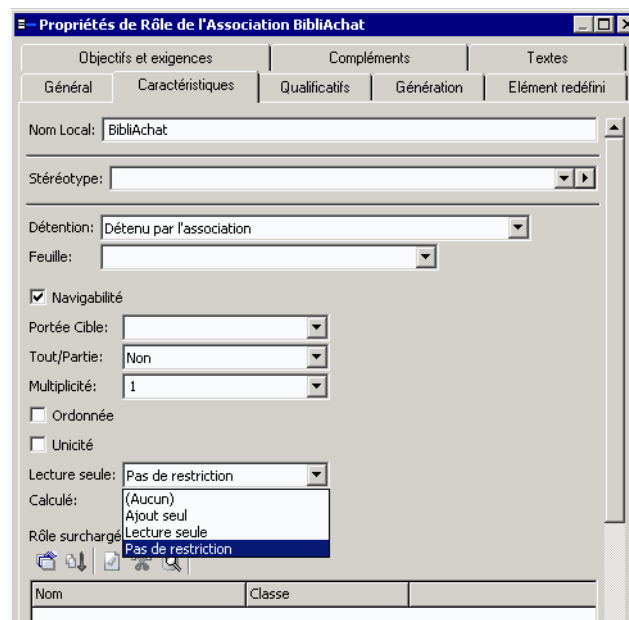
Rôle modifiable

La caractéristique **Lecture seule** permet de préciser si le rôle joué par une classe dans une association est modifiable après qu'il a été créé ou non. Par défaut, le rôle d'une classe dans une association est considéré comme modifiable.

Exemple : Une commande comprend une ligne de commande pour chacun des produits commandés. On ne peut plus changer ces différentes lignes de commande après l'enregistrement de la commande.



Il est possible d'indiquer si un rôle est modifiable à l'aide du menu contextuel du rôle ou dans la fenêtre de propriétés du rôle.



La caractéristique **Lecture seule** du rôle peut avoir les valeurs suivantes :

- **Ajout seul** : il est toujours possible de relier de nouveaux objets par cette association, mais il n'est pas possible de délier les objets déjà reliés.
- **Lecture seule** : les instances reliées ne peuvent plus être déliées. Il n'est pas possible non plus d'ajouter un nouveau lien.
- **Pas de restriction** : de nouvelles instances peuvent être reliées ou déliées à tout moment sans aucune contrainte.

Ordre d'un rôle

Il est possible de préciser si un rôle est ordonné ou non. Par exemple, pour une commande, il peut être intéressant de mémoriser l'ordre de ses lignes de commande.

Pour spécifier qu'un rôle est ordonné :

1. Ouvrez la fenêtre de **Propriétés** du rôle.
2. Dans l'onglet **Caractéristiques**, cochez la case **Ordonnée**.

Propriété statique d'un rôle

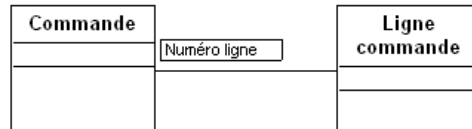
Comme pour un attribut, il est possible de préciser si un rôle peut prendre des valeurs spécifiques pour chacune des instances de la classe ou bien s'il a une valeur qui caractérise l'ensemble de la classe :

1. Ouvrez la fenêtre de propriétés du rôle.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Dans le champ **Statique**, sélectionnez :
 - "Oui" : pour que le rôle ait une valeur qui caractérise l'ensemble de la classe.
 - "Non" : pour que le rôle puisse prendre une valeur différente pour chacune des instances de la classe.

Qualificatif d'un rôle

Un qualificatif est un attribut dont les valeurs partitionnent l'ensemble des objets reliés à un objet à travers une association.

Exemple : Une commande comprend plusieurs lignes de commande. Le numéro de ligne de commande peut servir de qualificatif pour identifier chacune des lignes.

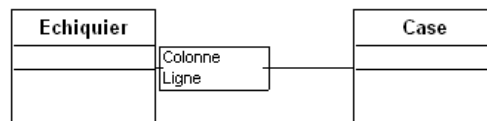


Pour renseigner un qualificatif :

1. Cliquez sur le rôle avec le bouton droit de la souris et sélectionnez **Propriétés**.
La fenêtre de propriétés du rôle s'ouvre.
2. Cliquez sur l'onglet **Qualificatifs**.
3. Pour ajouter un nouveau qualificatif au rôle, cliquez avec le bouton droit dans la fenêtre.
4. Dans le menu contextuel qui apparaît, activez la commande **Nouveau**.
Ce même menu contextuel vous permet de modifier les propriétés du qualificatif.

Plusieurs qualificatifs peuvent être nécessaires pour identifier de manière unique chacun des objets de la classe.

Par exemple, chaque case d'un échiquier est identifiée par son numéro de ligne et son numéro de colonne dans l'échiquier.



Surcharger un rôle

Un rôle peut hériter d'un rôle défini au niveau supérieur. La surcharge permet de définir des propriétés supplémentaires sur un rôle hérité.

Pour surcharger un rôle :

1. Ouvrez la fenêtre de propriétés du rôle.
2. Dans la fenêtre de propriétés, cliquez sur l'onglet **Caractéristiques**.
3. Dans le cadre **Rôle surchargé**, cliquez sur le bouton **Relier**.
La fenêtre de sélection apparaît.

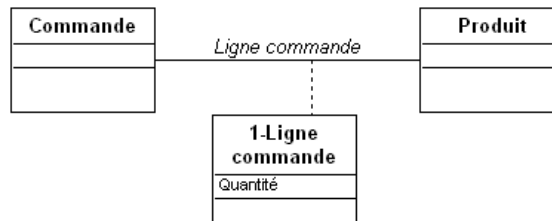
4. Sélectionnez **Propriétés surchargée**. Cette commande affiche les rôles qu'il est possible de surcharger.
5. Sélectionnez le rôle en question et cliquez sur **OK**.

Les classes d'association

Une classe d'association est une association qui possède aussi les propriétés d'une classe comme des attributs.

Il est utile de créer une classe d'association pour préciser des caractéristiques de l'association.

Par exemple, il est nécessaire de préciser la quantité de produit demandée pour chacune des lignes d'une commande.



Pour créer une classe d'association :

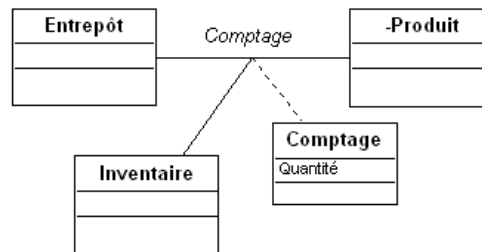
1. Créez une nouvelle classe.
2. A l'aide du bouton **Lien**, créez un lien entre la classe et l'association.
La classe d'association est reliée à l'association par un trait pointillé.

☛ Comme pour les classes standard, il est possible de cacher les compartiments et de retailler la classe d'association à l'aide de la commande **Affichage de son menu contextuel**.

Définir une association "plus que binaire"

Certaines associations associent non pas deux, mais davantage de classes. Ces associations sont, en principe, rares.

Exemple : Lors d'un inventaire, une certaine quantité de produit a été comptée dans chaque entrepôt.

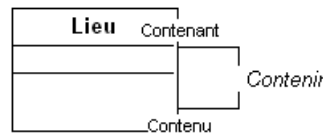


Pour créer une association ternaire :

1. Créez tout d'abord l'association entre deux des classes.
2. Cliquez sur le bouton **Rôle de l'association** .
3. Tirez un lien entre l'association et la troisième classe.

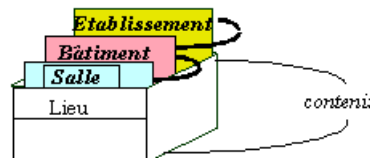
Vous pouvez ensuite créer la classe d'association éventuelle comme précédemment.

Les associations réflexives



Certaines associations mettent en jeu plusieurs fois la même classe.

Une salle de classe, un bâtiment, un établissement scolaire sont tous des lieux.




Une salle de classe est contenue dans un bâtiment, lui-même contenu dans un établissement scolaire.

Une association réflexive porte deux fois sur la même classe.

Créer une association réflexive

Pour créer une association réflexive :

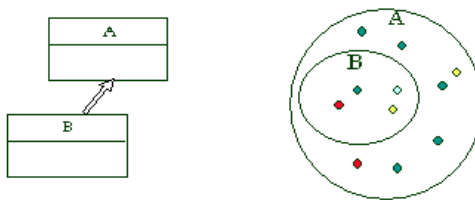
1. Cliquez sur le bouton **Association**  de la barre d'outils.
2. Cliquez dans la classe concernée et faites glisser la souris en dehors de cette classe, puis revenez-y ; relâchez enfin votre pression.
L'association réflexive apparaît sous forme d'un crochet.

☛ Dans le cas d'une association entre une classe et elle-même, il est indispensable de préciser les rôles afin de distinguer les liens correspondants dans le dessin.

LES GÉNÉRALISATIONS

Une généralisation représente une relation d'héritage entre une classe générale et une classe plus spécifique. La classe spécifique est cohérente avec la classe plus générale et en hérite ses caractéristiques et son comportement. Elle comporte cependant des informations supplémentaires. Toute instance de la classe spécifique est aussi une instance de la classe générale.

Qu'est-ce qu'une généralisation



La classe A est une généralisation de la classe B. Cela suppose que tous les objets de la classe B sont aussi des objets de la classe A. Autrement dit, B est un sous-ensemble de A.

B est alors la sous-classe, A la super-classe.

Exemple A : Personne, B : Parisien.

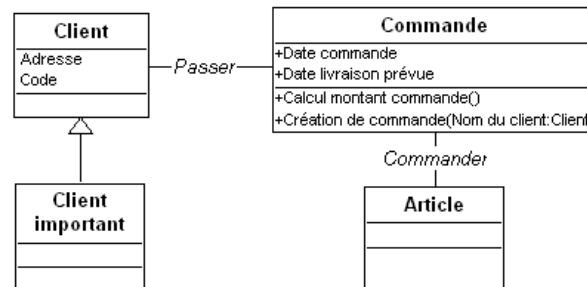
B étant un sous-ensemble de A, les objets de la classe B "héritent" des caractéristiques de ceux de la classe A.

Il n'est donc pas nécessaire de décrire de nouveau pour la classe B :

- Ses attributs
- Ses opérations
- Ses associations

Exemple

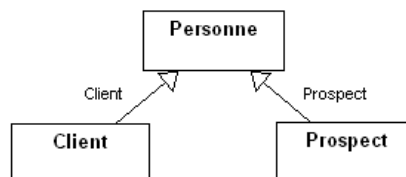
La classe "Client important" qui représente les clients dont le "C.A. sur les 12 derniers mois" dépasse 1 MF, peut être une spécialisation de la classe client (origine).



Dans l'exemple qui précède, les associations et les attributs spécifiés pour "Client" sont aussi valables pour "Client important".

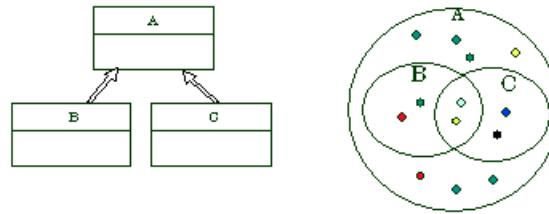
Autres exemples de généralisations :

- Prospect et client sont deux sous-classes de "personne".



- Commande export est une sous-classe de la classe "commande".
- Personne physique et personne morale sont deux sous-classes de la classe "personne".
- Polygone, ellipse et cercle sont des sous-classes de la classe "forme".
- Chêne, orme, et bouleau sont des sous-classes de la classe "arbre".
- Véhicule à moteur, véhicule tout-terrain et véhicule amphibie sont des sous-classes de la classe "véhicule".
- Camion est une sous-classe de la classe "véhicule à moteur".

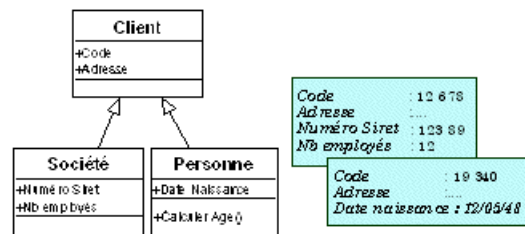
Cas de plusieurs sous-classes



Plusieurs sous-classes d'une même classe :

- Ne sont pas forcément exclusives.
- Ne forment pas nécessairement une partition.

Intérêt des sous-classes

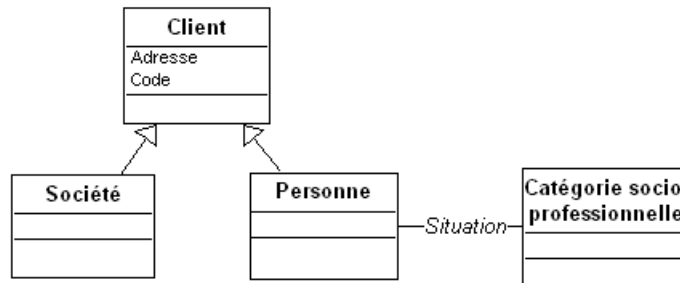


Une sous-classe hérite de tous les attributs, opérations et associations de sa super-classe, mais elle peut avoir des attributs ou des associations que ne possède pas sa super-classe.

Une sous-classe peut ainsi avoir des attributs spécifiques. Ceux-ci n'ont de sens que pour une sous-classe particulière. Dans l'exemple ci-dessus :

- Le "numéro de Siret" et le "nombre d'employés" n'ont de sens que pour une "société".
- La "date de naissance" est caractéristique d'une "personne", pas d'une "société".
- De même, il est utile de calculer l'"âge" d'une "personne". Cet attribut et cette opération n'ont généralement pas d'intérêt pour une "société".

Une sous-classe peut également avoir des *associations* spécifiques.



- Une "personne" entre dans une "catégorie socio-professionnelle" : "cadre", "employé", "commerçant", "agriculteur", etc. Cette classification n'a pas de sens pour une "société". (Il existe également une classification pour les entreprises, mais ce n'est pas la même que pour les personnes.)

Héritage Multiple

Il est parfois utile de spécifier une classe ayant plusieurs super-classes. La sous-classe hérite alors de toutes les caractéristiques des deux super-classes. Cette possibilité doit être utilisée avec précaution.

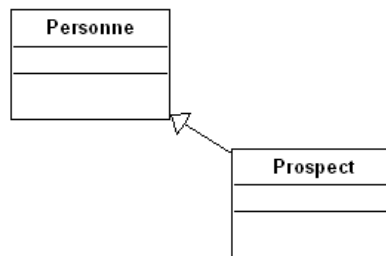
☛ L'héritage multiple n'est pas pris en compte pour la génération des tables.

Créer une généralisation

Pour créer une généralisation :

1. Cliquez sur le bouton **Généralisation**  de la barre d'outils
2. Cliquez dans la sous-classe et faites glisser la souris jusqu'à la super-classe, avant de relâcher votre pression.

La généralisation apparaît dans le diagramme, matérialisée par une flèche.

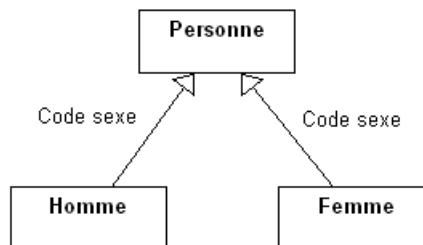


Vous pouvez également créer une généralisation à partir de l'éditeur de classes. Voir ["Créer des objets dans l'éditeur de classes", page 92.](#)

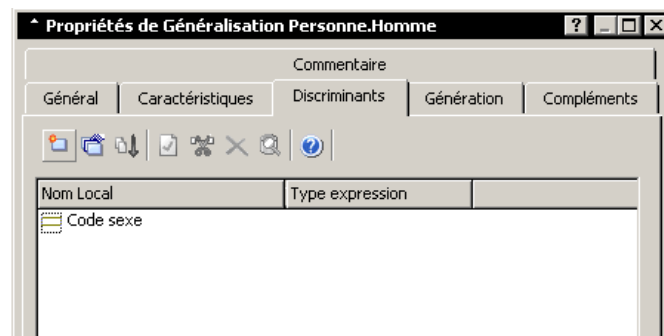
Discriminant

Le discriminant est l'attribut d'une généralisation dont la valeur permet de répartir les objets entre les sous-classes associées à la généralisation.

Par exemple, l'attribut code-sexe permet de répartir les objets de la classe personne entre la sous-classe homme ou femme.



Vous pouvez définir le ou les discriminants dans la fenêtre de propriétés de la généralisation.



Il est également possible de préciser si une généralisation est :

- *Disjointe* : Une instance ne peut pas appartenir simultanément à deux sous-classes de cette généralisation.
- *Complète* : Toutes les instances de la super-classe appartiennent à au moins une des sous-classes de cette généralisation.

SPÉCIFIER LES INTERFACES

Une interface représente la partie visible d'une classe ou d'un paquetage dans une relation contractuelle de type client - fournisseur. L'interface est un stéréotype de classe.

Une interface est constituée d'un ensemble d'opérations qui décrivent le comportement d'un élément. En particulier, une interface représente la partie visible d'une classe ou d'un paquetage dans une relation contractuelle de type client - fournisseur.

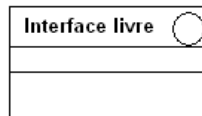
Ce sont des interfaces entre les différents composants du système informatique.

Créer une interface

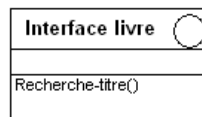
Pour créer une classe interface :

1. Sélectionnez le bouton **Interface**  dans la barre d'outils et cliquez dans le dessin.
2. Saisissez son nom dans la fenêtre qui apparaît.


La classe interface apparaît dans le dessin avec un rond distinctif en haut à droite.



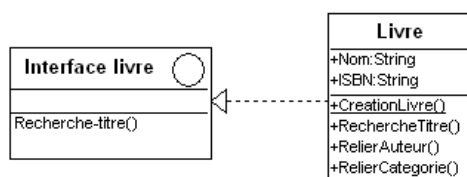
Vous pouvez spécifier les opérations de l'interface comme pour n'importe quelle classe.



Pour préciser qu'une interface est supportée par une classe :

1. Cliquez sur le bouton 


2. Effectuez le lien en partant de la classe fournisseur pour aller vers l'interface supportée.




Pour indiquer qu'une classe requiert une interface :


1. Cliquez sur le bouton
2. Effectuez le lien en partant de la classe cliente vers l'interface requise.

SPÉCIFIER LES DÉPENDANCES

Vous pouvez créer des paquetages à l'aide du bouton  de la barre d'objets.

☛ Le bouton **Vues**  vous permet de préciser les boutons que vous voulez voir apparaître dans la barre d'objets.

Pour indiquer qu'un paquetage référence une classe ou un autre paquetage :

1. Cliquez sur le bouton .
2. Effectuez le lien en partant d'un paquetage vers le paquetage ou la classe qu'il référence.

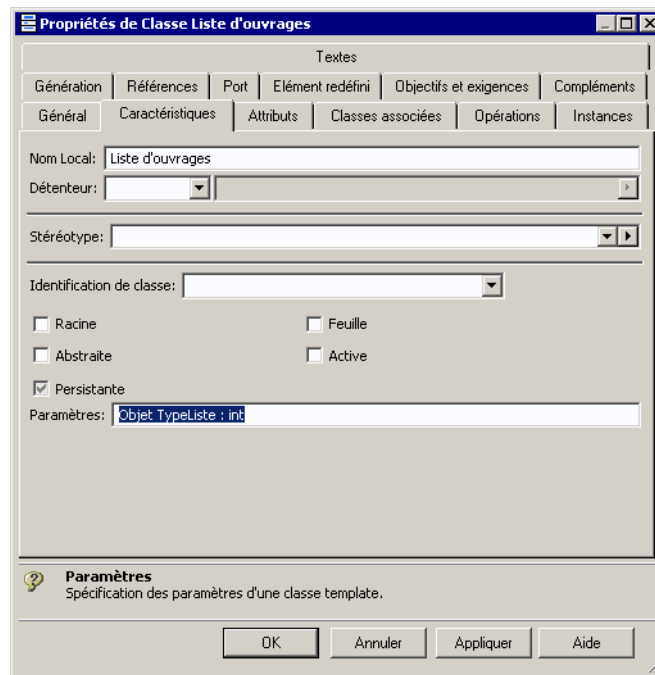
SPÉCIFIER DES CLASSES PARAMÉTRÉES

Une classe paramétrée permet de définir des caractéristiques et un comportement modulable en fonction de la valeur de certains paramètres. Par exemple, une classe paramétrée peut être utilisée pour gérer des listes d'objets. Le paramètre sera alors le type d'objet que l'on veut gérer sous forme de liste. Ce type de classe est implémenté en particulier dans le langage C++.

Spécifier une classe paramétrée


Pour spécifier une classe paramétrée :

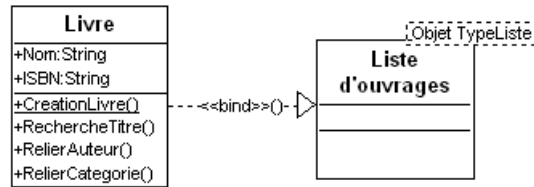
1. Ouvrez la fenêtre des **Propriétés** de la classe et cliquez sur l'onglet **Caractéristiques**.
2. Vous pouvez saisir les paramètres et préciser éventuellement leur type.



Les paramètres de la classe s'affichent en haut à droite.

Pour relier une classe à une classe paramétrée :

- Cliquez sur le bouton 
- Créez le lien en partant de la classe pour aller vers la classe paramétrée.



LES CONTRAINTES

Une contrainte est une déclaration qui établit un contrôle ou une règle de gestion impliquant généralement plusieurs classes.

La plupart des contraintes impliquent les associations entre les classes.


Exemples de contraintes :

- La personne responsable d'un service doit appartenir à ce service.
- Toute commande facturée doit avoir été livrée auparavant.
- La date de livraison doit être postérieure à la date de commande.

Un capteur couvrant une zone ne peut déclencher qu'une alarme protégeant cette même zone.

Pour créer une contrainte :

1. Cliquez sur le bouton **Contrainte**  de la barre d'objets.



 *S'il n'est pas affiché, cliquez sur le menu **Affichage** > **Vues et détails** et cochez la case "Contraintes".*

2. Cliquez dans une des associations concernées par la contrainte, et faites glisser la souris jusqu'à la deuxième association, avant de relâcher votre pression.

La fenêtre d'ajout d'une contrainte s'ouvre.

3. Saisissez le nom de la contrainte puis cliquez sur **Créer**.

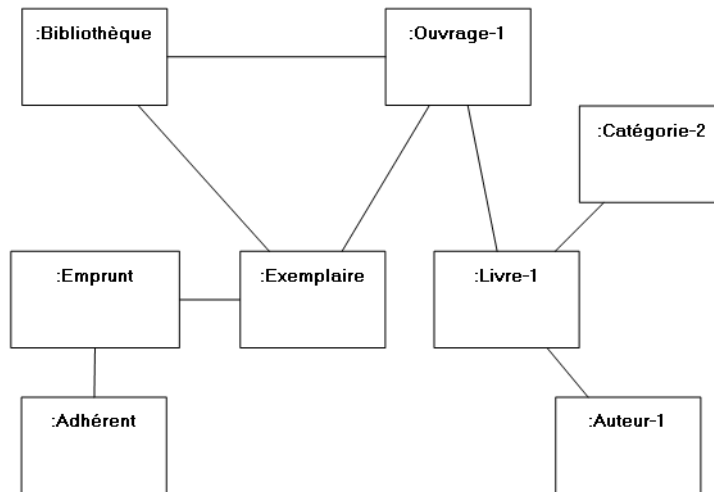
La contrainte apparaît dans le diagramme.

 *Vous pouvez relier une contrainte à d'autres classes ou associations à l'aide du bouton **Lien** .*

LE DIAGRAMME D'OBJETS

Un diagramme d'objets, ou diagramme d'instances, contient des objets avec des valeurs exemples pour leurs attributs, et des liens. Il montre en détail l'état du système à un instant précis.

Vous pouvez créer le diagramme d'objets d'une classe, d'un composant, d'un paquetage ou d'un cas d'utilisation.



Les objets

Un objet est une entité avec une identité et des frontières clairement définies dont l'état et le comportement sont encapsulés. Son état est défini par les valeurs de ses attributs et de ses liens avec d'autres objets. Son comportement est représenté par ses opérations et ses méthodes. Un Objet est une instance de Classe.


Exemples d'objets :

- Objets de gestion :
 - Jacques Dupond, Pierre Durand, Paul Smith sont des instances de la classe personne.
 - Les commandes no 10533 et 7322 sont des instances de la classe commande.
 - Ecran Sony SPD-1730, Compaq Deskpro 200 sont des instances de la classe article.
 - Dupond de Nemours, Burger King sont des instances de la classe société.
- Objets techniques utilisés pour la programmation :
 - Dlg_Order_Create, Dlg_Customer_Query sont des instances de la classe fenêtre.
 - Str_Customer_Name, Str_Product_Comment sont des instances de la classe chaîne.

☛ Les objets représentés dans un diagramme d'objets peuvent être des instances de classe, de paquetage, de cas d'utilisation, de composant ou de nœud, ce qui permet de définir des diagrammes de séquence au niveau de détail souhaité.

Créer un objet (une instance)

Pour créer un objet :

1. Cliquez sur le bouton  **Instance**.
Vous pouvez créer des objets de différents types. La flèche située à droite du bouton offre un raccourci vers les types d'objets Classe ou Composant, plus fréquemment utilisés.
2. Puis cliquez sur le plan de travail du diagramme.
La fenêtre permettant d'ajouter d'une instance s'ouvre.
3. Saisissez le **Nom** de l'instance.
4. Précisez si nécessaire le **Type d'instance**.
5. Cliquez sur **Créer**.

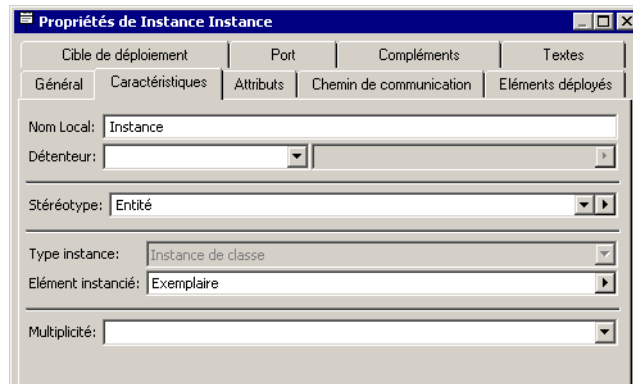
L'instance est posée dans le diagramme.

Propriétés d'une instance

Pour ouvrir la fenêtre de propriétés d'une instance :

- Cliquez avec le bouton droit sur l'instance et sélectionnez **Propriétés**.

Plusieurs onglets permettent de définir les propriétés d'une instance.



Vous pouvez :

- Sélectionner le **Type** de l'instance (Acteur, Classe, etc.).
- Préciser de quelle **Classe**, quel **Acteur**, etc. cet objet est une instance.
- Indiquer un nom pour cette Instance.
- Préciser son **Stéréotype**.

Valeur d'un attribut

Pour renseigner la valeur d'un attribut :

1. Cliquez avec le bouton droit sur l'instance de la classe contenant l'attribut.
2. Sélectionnez **Attributs**.
3. Dans la fenêtre qui apparaît, indiquez la valeur de l'attribut. Vous pouvez renseigner une valeur instanciée ou une valeur constante.
 - **Valeur instanciée** : cliquez dans cette colonne pour afficher la liste des instances possibles pour l'attribut sélectionné. Il s'agit de valeurs variables.
 - **Valeur** : cliquez dans la colonne et entrez la valeur de l'attribut.

Les liens


Un lien entre objets représente une instance d'association entre deux objets.

Exemples de liens entre objets :

- La commande n° 10733 a été passée par Jacques Dupond.
- La commande 10733 comprend les produits Ecran Sony SPD-1730 et Compaq Deskpro 200.
- Mr Jacques Dupond travaille pour la société Dupond de Nemours.
- La fenêtreDlg_Customer_Query affiche la chaîne de caractères Str_Customer_Name.

Créer un lien

Pour créer un lien :

1. Cliquez sur le bouton **Lien**  de la barre d'outils.
2. Cliquez dans un des objets concernés, et faites glisser la souris jusqu'au deuxième objet, avant de relâcher votre pression.

Le lien apparaît alors dans le dessin.

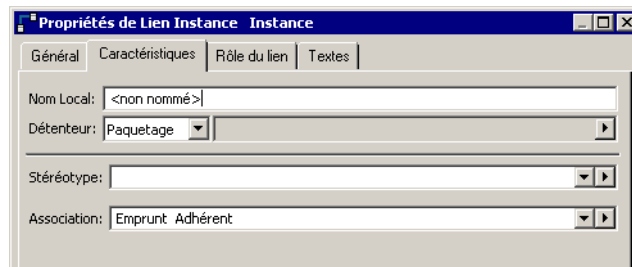
Si un lien entre les deux objets existe déjà, une fenêtre s'ouvre pour vous permettre de choisir parmi les liens existants ou en créer un nouveau.

Propriétés d'un lien

Pour ouvrir la fenêtre de propriétés d'un lien :

- Cliquez avec le bouton droit au centre du lien et sélectionnez **Propriétés**.

La fenêtre de propriétés s'affiche.



☛ Si vous ne cliquez pas au centre du lien, c'est la fenêtre **Propriétés d'un des rôles** qui va s'ouvrir.

Sous l'onglet **Caractéristiques**, vous pouvez préciser :

- Le **Nom** du lien.
- Le **Stéréotype** du lien.
- L'**Association** correspondant au lien.
- Le **Paquetage** détenteur du lien.

Et sous l'onglet **Rôle du lien** :

- Pour chaque **Instance** reliée par ce lien, le nom du **Rôle** et la **Multiplicité** de ce rôle.

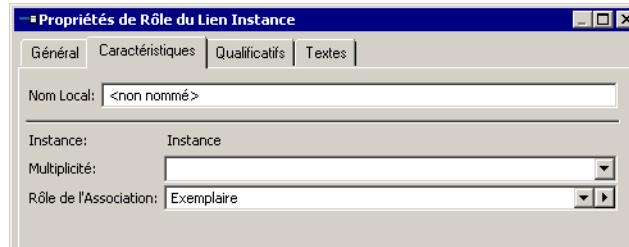
☛ Parmi les associations proposées ne figurent que celles qui figurent entre les classes des deux objets.

Propriétés d'un rôle

Pour ouvrir la fenêtre de propriétés d'un rôle :

1. Dans la fenêtre de propriétés d'un lien, cliquez sur l'onglet **Rôle du lien**.
2. Cliquez avec le bouton droit sur le rôle et sélectionnez la commande **Propriétés**.

La fenêtre de propriétés du rôle s'affiche.



☛ Si vous ne cliquez pas sur le rôle, c'est la fenêtre de propriétés du lien qui va s'ouvrir.

Dans cette fenêtre vous pouvez préciser :

- Un **Nom** pour l'instance de rôle.
- Le **Rôle** de cette instance.
- La **Multiplicité** de l'instance de rôle.
- Pour cette instance de rôle, les valeurs des *qualificatifs* définis au niveau de la classe.

L'ÉDITEUR DE CLASSES

L'éditeur de classes vous permet de visualiser les propriétés d'un paquetage ou d'une classe ainsi que ses liens dans la base.

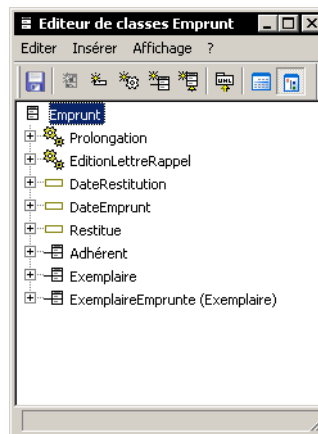
Ouvrir l'éditeur de classes

L'éditeur de classes est accessible à partir d'une classe ou d'un paquetage, et ce depuis le navigateur ou depuis un diagramme.

Pour ouvrir l'éditeur de classes :

1. Cliquez avec le bouton droit sur le paquetage ou la classe.
2. Sélectionnez **Editeur de classes**.

L'éditeur de classes s'affiche. Il présente sous forme d'arborescence le paquetage ou la classe en question ainsi que les objets associés.



Paramètres d'affichage de l'éditeur de classes

Pour personnaliser l'affichage de l'éditeur de classes :

- Cliquez sur le menu **Affichage** > **Barre d'outils...**
- Dans la fenêtre qui apparaît, cochez la case **Barre de filtrage**.

Les boutons apparaissant dans la barre d'outils vous permettent d'afficher :



les attributs



les opérations



les interfaces supportées



les classes héritées



les classes héritières

Pour conserver l'éditeur au premier plan :

- Cliquez sur le bouton

Pour actualiser l'affichage :

- Cliquez sur le bouton

☛ Ces boutons peuvent être filtrés. Dans ce cas, cliquez sur le menu **affichage** > **Barre d'outils...**

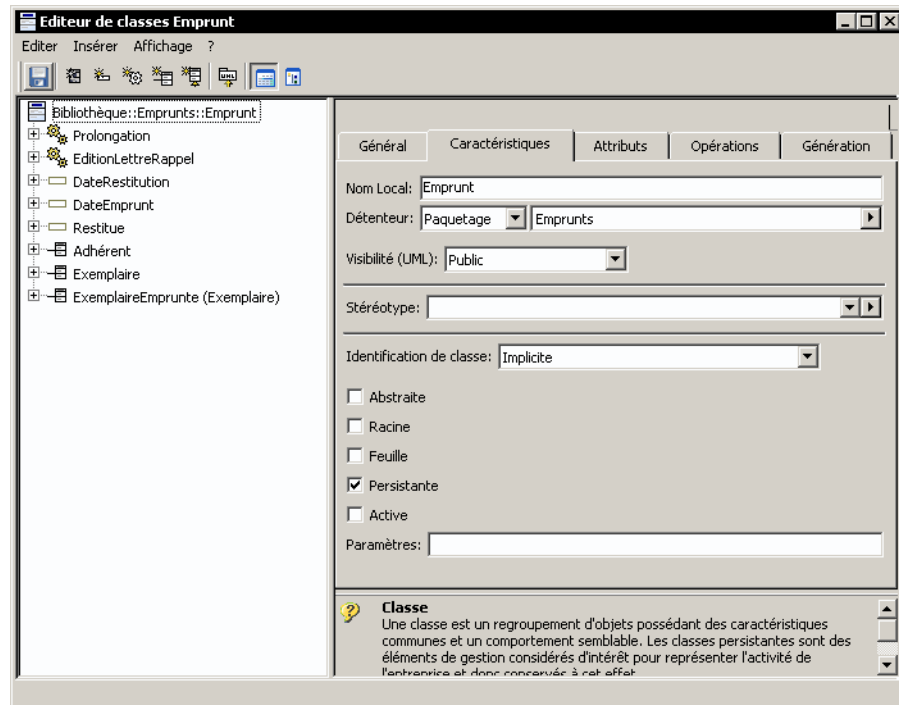
Propriétés des objets

Vous pouvez afficher le détail des objets présentés dans l'arborescence.

Pour voir le détail des objets :

- Cliquez sur le bouton **Détails** de l'éditeur.

Les propriétés de l'objet sélectionné s'affichent dans la partie droite de la fenêtre.




Tout comme dans la fenêtre de propriétés, différents onglets vous permettent de visualiser et de modifier les propriétés de l'objet.

➡ Voir "*Propriétés d'une classe*", page 41.

Créer des objets dans l'éditeur de classes

L'éditeur de classes vous permet de créer des attributs, des opérations, des associations et des héritages.

Par exemple, pour créer une opération :


1. Cliquez dans l'arborescence sur la classe concernée.
2. Cliquez sur le bouton **Création d'une opération**  dans la barre d'outils de l'éditeur.
La fenêtre **Création d'une opération** apparaît.
3. Entrez le nom de l'opération puis cliquez sur **OK**.


L'opération créée apparaît à la fois dans l'arborescence et sous l'onglet **Opération** de la fenêtre de propriétés de la classe.

Créer une association entre deux classes

Pour créer une association entre deux classes :

1. Sélectionnez dans l'éditeur la classe pour laquelle vous voulez créer une association.


 Vous pouvez également créer une association à partir d'un rôle.

2. Cliquez sur le bouton **Création d'une association**  de la barre d'outils.
3. Dans la fenêtre qui apparaît, saisissez le nom de la classe associée. Vous pouvez rechercher une classe existante ou en créer une nouvelle.
4. Cliquez sur **OK**.

L'association apparaît dans l'éditeur.

Créer une généralisation entre deux classes

Pour créer une généralisation entre deux classes :

1. Sélectionnez dans l'éditeur la classe pour laquelle vous voulez créer une généralisation.
2. Cliquez sur le bouton **Création d'un héritage**  de la barre d'outils.
3. Dans la fenêtre qui apparaît, saisissez le nom de la classe dont vous voulez hériter. Vous pouvez rechercher cette classe en cliquant sur la flèche noire située à l'extrémité du champ.
4. Cliquez sur **OK**.

La sur-classe apparaît dans l'éditeur.

GÉNÉRER UN DIAGRAMME DE CLASSES

Vous pouvez générer un diagramme de classes depuis :

- Un paquetage
- Une classe

Depuis un paquetage

- Cliquez avec le bouton droit sur le paquetage.
- Sélectionnez **Nouveau > Générer un diagramme de classes**.

Cette commande construit un diagramme descriptif du paquetage. Il contient toutes les classes détenues par le paquetage, et pour chaque classe :

- Un niveau d'héritage.
- Le premier niveau d'association avec les éventuelles classes associatives.
- Le premier niveau d'interfaces implémentées.

Depuis une classe

Vous pouvez générer deux types de diagrammes de classes :

- **Diagramme de classes simple** : il décrit pour la classe en question le premier niveau de classes associées avec les éventuelles classes associatives, le premier niveau d'héritage et le premier niveau d'interfaces implémentées.
- **Diagramme de classes élaboré** : il décrit pour la classe en question *n* niveaux de classes associées avec les éventuelles classes associatives (la récursion s'arrête si une classe C n'appartient pas au même paquetage que la classe décrite mais si C est incluse dans le diagramme), le premier niveau d'héritage et le premier niveau d'interfaces implémentées.

Fonctionnalité de réorganisation automatique

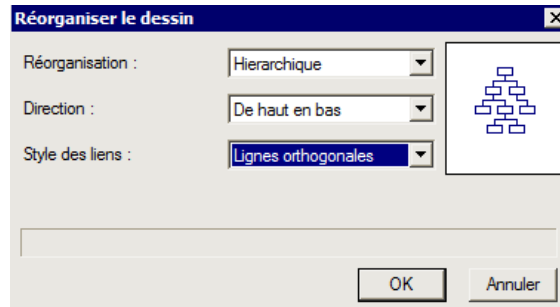
Le graphe des utilisateurs, le diagramme de classes, le diagramme de collaboration, le diagramme de composants et le diagramme relationnel comportent la fonctionnalité de réorganisation du dessin.

☺ *La fonctionnalité de réorganisation est déclenchée automatiquement au chargement d'un diagramme qui ne comporte pas encore de dessin.*

Pour modifier l'organisation d'un dessin existant :

1. Sélectionnez le menu **Dessin > Réorganiser le dessin**.

2. Sélectionnez le mode de réorganisation souhaité, la direction ainsi que le style des liens dans le diagramme.



😊 L'image miniature à côté des options de réorganisation vous permet d'avoir un aperçu de chaque type de réorganisation.

3. Cliquez sur **OK** pour appliquer les modifications.

LES DIAGRAMMES DE STRUCTURE ET DE DÉPLOIEMENT



Outre les diagrammes de classes et d'objets, les diagrammes structurels comprennent :

- ✓ Le diagramme de paquetages, qui permet d'organiser les éléments du modèle.
- ✓ Le diagramme de composants, qui met en évidence les relations de dépendance entre composants.
- ✓ Le diagramme de structure composite, qui décrit les interactions entre les composants et leurs parties.

LE DIAGRAMME DE PAQUETAGES

Un diagramme de paquetages permet d'organiser les éléments de modélisation, de manière à assurer une partition des travaux de spécifications et de développement.

Un élément ne doit apparaître que dans un seul paquetage.

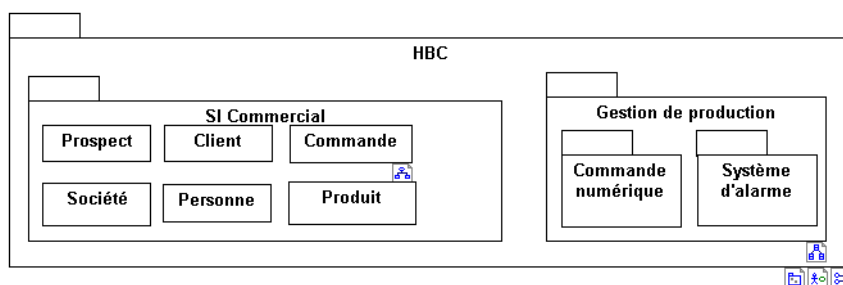
Le découpage en paquetage est généralement fait de manière à minimiser les interactions entre les différents paquetages.

Exemple de diagramme de paquetages

Le paquetage "HBC" contient les paquetages "SI Commercial" et "Gestion de production".

Le paquetage "Gestion de production" se décompose en deux paquetages "Commande numérique" et "Système d'alarme".

Le paquetage "SI Commercial" contient les classes "Prospect", "Client", "Société", "Personne", "Commande" et "Produit".




Créer un diagramme de paquetages

Un diagramme de paquetage se crée depuis un paquetage. Pour la création d'un paquetage, voir "[Créer un diagramme de cas d'utilisation](#)", page 20.

Pour créer un diagramme de paquetages :

1. Dans la fenêtre de navigation **Objets principaux**, cliquez avec le bouton droit sur le nom du paquetage.
2. Dans le menu contextuel qui s'affiche, cliquez sur **Nouveau > Diagramme**.
Une fenêtre contenant les différents types de diagramme possibles apparaît.
3. Sélectionnez "Diagramme de paquetages" et cliquez sur le bouton **Créer**.

Les paquetages


 Un **paquetage** partitionne le domaine d'étude et les travaux associés. Il permet de regrouper divers éléments, en particulier des cas d'utilisations et des classes. Un **paquetage** peut aussi contenir d'autres **paquetages**. Les **paquetages** sont liés entre eux à travers des rapports contractuels définissant leur interface.

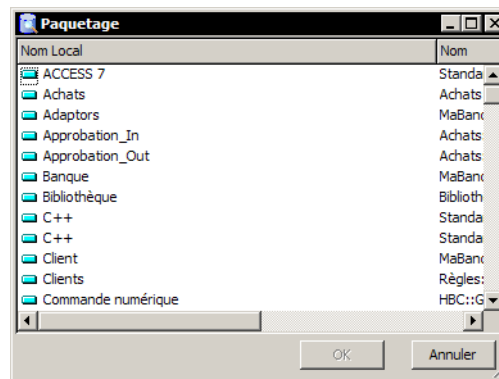
Exemples de **paquetages** :

- Le système d'information commercial.
- La comptabilité.
- La gestion de production.
- La commande numérique d'une machine.
- La gestion des stocks.
- La gestion du système d'alarme et du téléphone.

Retrouver un paquetage existant

Dans le diagramme de paquetages :

1. Cliquez sur le bouton **Paquetage** de la barre d'objets puis cliquez sur le plan de travail.
2. Dans la fenêtre **Ajout d'un paquetage**, sélectionnez la commande **Lister** à l'aide de la flèche .
La liste des paquetages apparaît.



😊 Notez que vous pouvez faire apparaître la liste des paquetages existants en utilisant la touche <Ctrl-L>.

Vous pouvez également saisir le début du nom, par exemple "Paqu", et en utilisant la touche <Ctrl-L>, faire apparaître la liste des paquetages dont le nom commence par "Paqu".

3. Sélectionnez le paquetage qui vous intéresse et cliquez sur **OK**.
Le nom du diagramme apparaît dans la fenêtre **Ajout d'un paquetage**.
4. Cliquez sur **Relier**.


Le paquetage apparaît dans le diagramme.

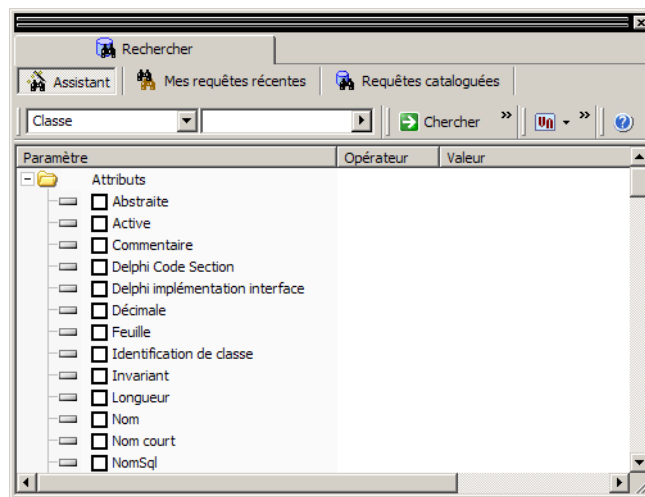
Les classes

Le diagramme de paquetages permet de répartir les classes entre les paquetages.

Retrouver les classes existantes

Pour ajouter les classes dans le diagramme de paquetages :

1. Recherchez les classes déjà créées en cliquant sur le bouton de recherche  de la barre **MEGA**.
2. Dans la fenêtre qui s'ouvre, choisissez la cible "Classe".




3. Cliquez sur **Chercher**.
La liste des classes s'ouvre.
4. Sélectionnez les classes qui vous intéressent et glissez-les dans le diagramme.

Spécifier les dépendances dans un diagramme de paquetage

Des liens vous permettent d'indiquer si un paquetage détient ou référence une classe ou un autre paquetage.


Pour indiquer qu'un paquetage référence une classe ou un autre paquetage :

1. Cliquez sur le bouton .
2. Puis, effectuez le lien en partant d'un paquetage vers le paquetage ou la classe qu'il référence.
Une fenêtre vous demande le type de lien à créer.
3. Sélectionnez "Paquetage référencé" ou "Classe référencée" selon qu'il s'agisse d'un paquetage ou d'une classe.

LE DIAGRAMME DE COMPOSANTS

Un diagramme de *composants* présente l'interdépendance des composants logiciels et des *interfaces* (il définit qui utilise quoi).

 *Un composant représente une partie modulaire d'un système qui encapsule son contenu et qui est remplaçable dans son environnement. Un composant définit son comportement par les interfaces qu'il fournit et celles qu'il requiert.*

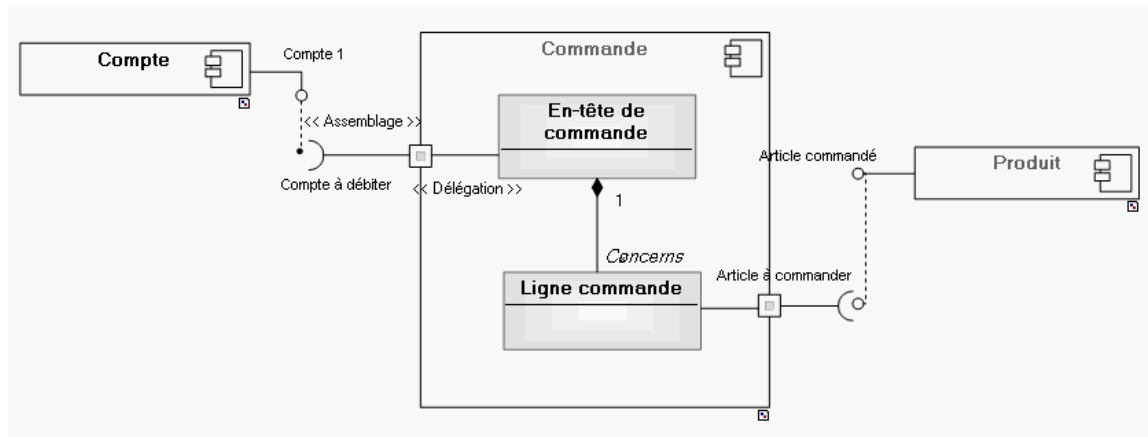
 *Une interface représente la partie visible d'une classe ou d'un paquetage dans une relation contractuelle de type client - fournisseur. L'interface est un stéréotype de classe.*

Un diagramme de composants contient des composants et des classes de stéréotype interface. Il est également possible d'y préciser les paquetages implémentés par les composants.

Vous pouvez créer un diagramme de composants depuis un composant ou un paquetage.

Exemple de diagramme de composants

Ce diagramme décrit les éléments détenus par le composant "Commande" et les interactions de ces éléments avec des composants externes.




Les composants

Un composant représente une partie modulaire d'un système qui encapsule son contenu et qui est remplaçable dans son environnement. Un composant définit son comportement par les interfaces qu'il fournit et celles qu'il requiert.

Un composant peut être remplacé par un autre si leurs interfaces sont conformes.

Un composant peut être un logiciel, un programme, un élément de code, etc.

Il est représenté par l'icône suivante : 

Les interfaces


Créer les interfaces des composants

Une interface représente la partie visible d'une classe ou d'un paquetage dans une relation contractuelle de type client - fournisseur.

L'interface est un type particulier de classe.

Pour créer une classe de stéréotype "Interface" dans le diagramme de structure composite :

1. Cliquez sur le bouton **Interface**  puis cliquez dans le diagramme.
2. Dans la fenêtre qui apparaît, saisissez le nom de la classe.
3. Cliquez sur **Créer**.

 Vous pouvez spécifier le détail de l'interface en termes d'attributs et d'opérations dans le diagramme de classes de la même manière que pour une classe.

Relier les interfaces aux autres objets

Deux types de lien permettent de différencier les interfaces requises des interfaces fournies.

Une interface requise est une interface nécessaire au fonctionnement de l'objet.


Exemple : le composant « Gestion des achats » a besoin pour son fonctionnement de l'interface « Produit » pour pouvoir associer une commande d'achat aux produits commandés.

Une interface fournie est une interface mise à disposition par un objet à destination d'autres objets.


Exemple : le composant « Gestion des produits » met à disposition l'interface « Produit ».

Vous pouvez définir les interfaces requises et les interfaces fournies par un objet indépendamment des autres objets.

Pour préciser qu'une interface est supportée par un objet :


1. Cliquez sur le bouton 
2. Dessinez le lien en partant de l'objet fournisseur (un composant, un paquetage ou une classe) pour aller vers l'interface supportée.

Pour indiquer qu'un objet requiert une interface :

1. Cliquez sur le bouton 

2. Dessinez le lien en partant de l'objet client vers l'interface requise.

Vous pouvez également spécifier les dépendances entre

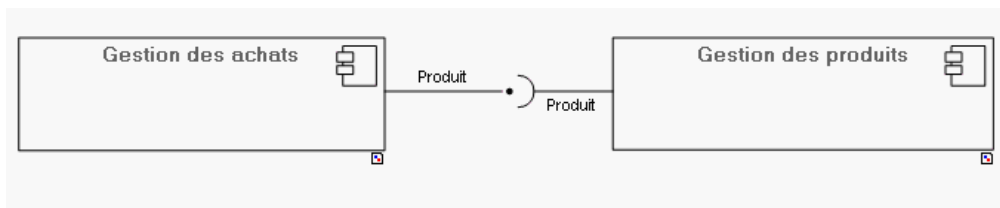
paquetages ou entre paquetages et classes à l'aide du bouton  comme dans le diagramme de paquetages.

Selon le type de lien, la forme de l'interface change : l'interface requise est représentée par un demi-cercle, l'interface fournie est représentée par un cercle.

Relier des interfaces

Deux interfaces peuvent être reliées l'une à l'autre. Cette connexion est modélisée par un connecteur.

Vous pouvez également indiquer qu'une interface fournie par un objet est requise par un autre. Il s'agit ici d'une seule et même interface.



Les ports

Les ports permettent de connecter un composant à ses parties ou à son environnement.

Les ports sont symbolisés par un carré dans le diagramme, et posés en bordure du composant décrit lorsqu'ils assurent la connexion avec l'extérieur.

Ils sont reliés aux composants par des connecteurs.

Associer une interface à un port

Les ports peuvent spécifier les requêtes envoyées et les services fournis par le composant ainsi que les requêtes et services qu'ils peuvent requérir d'autres parties du système. Ces requêtes et services sont représentés par des classes de type Interface.

Vous pouvez visualiser les interfaces associées à un port dans la fenêtre de propriétés d'un port, sous l'onglet **Interfaces fournies et requises**.

Les connecteurs

Les connecteurs permettent de relier les objets du diagramme.

Les connecteurs de type simple ne spécifient aucun type de connexion particulier, ils sont utilisés notamment pour relier les instances d'objets décrits dans des collaborations.

Dans le diagramme de structure composite, il est possible de spécifier le type de connecteur qui relie deux composants : Assemblage ou Délégation.

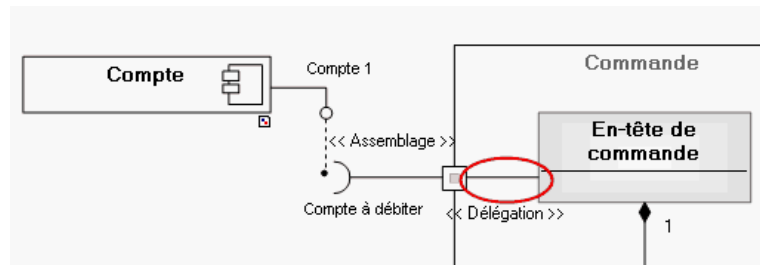
Connecteur de délégation

Un connecteur de type "Délégation" montre le réacheminement de requêtes vers un élément du composant chargé de les réaliser.

Le lien de délégation peut se faire directement entre le port du composant et l'élément du composant ou entre le port du composant et le port de l'élément.

Exemple

Ci-dessous, le composant "Commande" délègue la gestion des comptes à débiter à la classe "En-tête de commande".



Connecteur d'assemblage

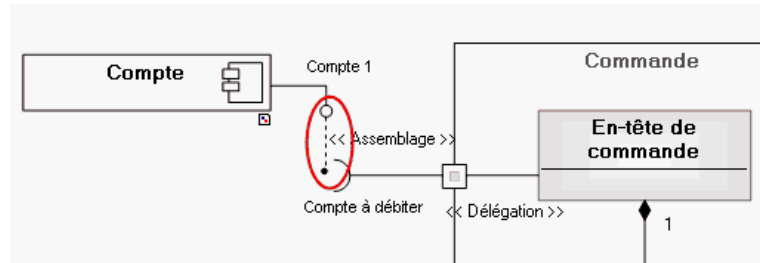
Un connecteur de type "Assemblage" est un connecteur entre deux ou plusieurs composants ou ports qui indique qu'un ou que plusieurs composants fournissent les services que d'autres utilisent.

☛ Il peut s'agir d'autres objets que de composants.

Pour relier des ports ou des composants qui partagent une interface, vous pouvez également utiliser les liens "Interface fournie" et "Interface requise".

Exemple

Un connecteur de type "Assemblage" relie l'interface fournie par le composant "Compte" à l'interface requise par la classe "En-tête Commande".



LE DIAGRAMME DE STRUCTURE COMPOSITE

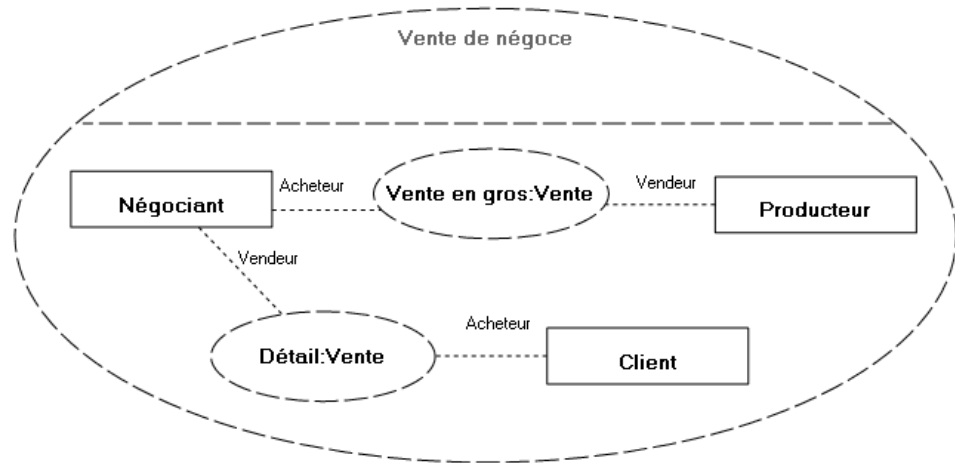
Le diagramme de structure composite permet de décrire la structure interne d'un composant, d'un paquetage ou d'une classe structurée.

Il permet également de préciser les collaborations qui interviennent entre les éléments de la structure dans l'exécution d'une tâche, en mettant en évidence le rôle joué par chaque élément dans la collaboration.

Les éléments de ce diagramme sont les parties (les *parts* en anglais), les ports par les biais desquels les parties interagissent avec l'extérieur, et les connecteurs reliant les parties entre elles ou avec les ports.

Exemple de diagramme de structure composite

Ce diagramme décrit le rôle joué par les parties dans la collaboration "Vente de négoce".



Les parties

Une partie représente un rôle joué par une instance d'une classe ou d'un composant lors de l'exécution d'une tâche.

Les parties sont reliées entre elles par des connecteurs ou des dépendances.

Une partie peut également être reliée - via un connecteur- à un port qui assure l'interface entre le composant décrit et l'extérieur.

Pour plus de détails sur ces éléments, voir :

- ✓ "Les connecteurs", page 103
- ✓ "Les liens de dépendance", page 108
- ✓ "Les ports", page 103.

Multiplicité


Les multiplicités définies sur les parties indiquent le nombre d'instances qui sont créées. Les multiplicités sur les rôles de connecteur indiquent le nombre de liens qui peuvent être créés pour chacune de ces instances.

Pour définir la multiplicité d'une partie :

1. Ouvrez la fenêtre de propriétés de la partie.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Cliquez sur la flèche située à l'extrémité du champ **Multiplicité** et sélectionnez la multiplicité voulue.
4. Cliquez sur **OK**.


Les collaborations

Dans le diagramme de structure composite, une *collaboration* décrit les rôles joués par chaque partie (instance) dans la réalisation d'une tâche.

 Une collaboration (UML) décrit une structure collaborative entre plusieurs éléments (rôles), qui accomplissent chacun une fonction spécialisée et qui réalisent collectivement une fonctionnalité attendue du système. Son objectif est de montrer comment un système fonctionne indépendamment d'une utilisation spécifique. On en retirera donc généralement l'identité précise des classes ou des instances qui y participent.

Elle est représentée par un ovale en pointillé contenant les instances de la collaboration.

Ces instances sont reliées entre elles par des *connecteurs*. A chaque extrémité du connecteur s'affiche le rôle qui correspond au nom de l'instance.


 Un connecteur est un lien qui permet d'établir une communication entre plusieurs objets. Un connecteur de délégation relie le contrat externe de l'objet (tel qu'il est spécifié par ses ports et/ou ses interfaces) aux objets internes qui vont le réaliser. Un connecteur d'assemblage entre plusieurs objets (ou leurs ports) permet de spécifier comment un des objets fournit l'interface requise par un autre.

Il est possible d'appliquer le modèle d'une collaboration à différentes instances.

Utilisation de collaboration

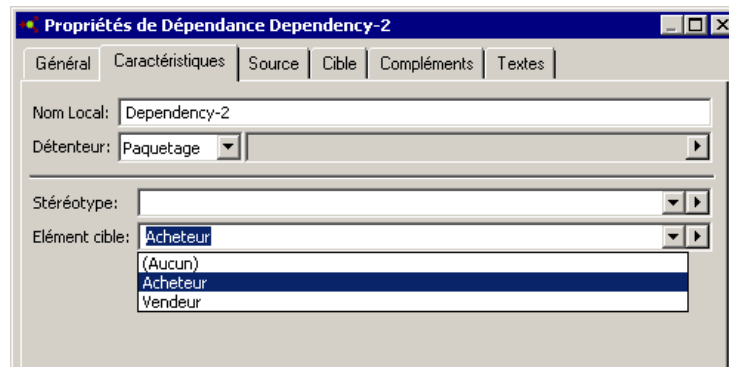
Une utilisation de collaboration représente l'application de la structure décrite par une collaboration à une situation particulière mettant en oeuvre des classes ou des instances spécifiques. Ces classes ou ces instances jouent alors les rôles définis dans la collaboration.

Les instances sont reliées à l'utilisation d'une collaboration par un lien de *dépendance* sur lequel doit être précisé le rôle joué par l'instance.

 Une dépendance précise que l'implémentation ou le fonctionnement d'un ou de plusieurs éléments nécessite la présence d'un ou de plusieurs autres éléments. Il existe plusieurs stéréotypes de dépendance.

Exemple d'utilisation de collaboration

Dans le cas d'une demande d'achat entre deux instances d'acteur, une collaboration est utilisée. Cette collaboration relie deux rôles : le rôle d'acheteur et le rôle de vendeur. Sur la dépendance qui relie chaque instance à la collaboration, vous pouvez indiquer le rôle que joue l'instance.



Les liens de dépendance

Une dépendance précise que l'implémentation ou le fonctionnement d'un ou de plusieurs éléments nécessite la présence d'un ou de plusieurs autres éléments.

Une dépendance est également une relation de type fournisseur / client qui indique quel est l'élément source et l'élément cible dans la collaboration.




Nature de la dépendance

Un stéréotype sur la dépendance permet de spécifier la nature de la dépendance :

- **Binding** : le binding est une relation entre un template et un élément de modélisation généré à partir du template. Il inclut une liste d'arguments en correspondance avec les paramètres du template.
- **Derive** : indique une relation de dérivation entre des éléments de modélisation qui sont généralement, mais pas nécessairement, de même type. Une telle relation de dépendance implique que l'un des éléments peut être calculé à partir de l'autre.
- **Mapping UML/XML** : une expression de mapping qui définit la relation entre les éléments (classes, attributs, ...) d'un schéma ou d'un diagramme de classes et ceux d'un autre schéma ou diagramme de classes.
- **Refine** : spécifie une relation de dépendance entre des éléments de modélisation à différents niveaux sémantiques, tels que l'analyse et la conception.
- **Trace** : spécifie une relation de traçabilité entre des éléments de modélisation ou des ensembles d'éléments de modélisation qui représentent le même concept dans différents modèles.

Pour préciser la nature d'une dépendance :

1. Ouvrez la fenêtre de propriétés de la dépendance.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Dans le champ **Stéréotype**, déroulez la liste et sélectionnez l'un des stéréotypes proposés.

La flèche  vous permet également de créer de nouveaux stéréotypes.

LE DIAGRAMME DE MACHINE À ÉTATS



Un diagramme de machine à état permet de décrire les comportements possibles d'un objet, suivant les événements auxquels il est soumis au cours de son cycle de vie.

Les points suivants sont abordés ici :

- ✓ ["Présentation du diagramme de machine à états", page 112](#)
- ✓ ["Créer un diagramme de machine à états", page 112](#)
- ✓ ["Les états", page 113](#)
- ✓ ["Les transitions entre états", page 117](#)

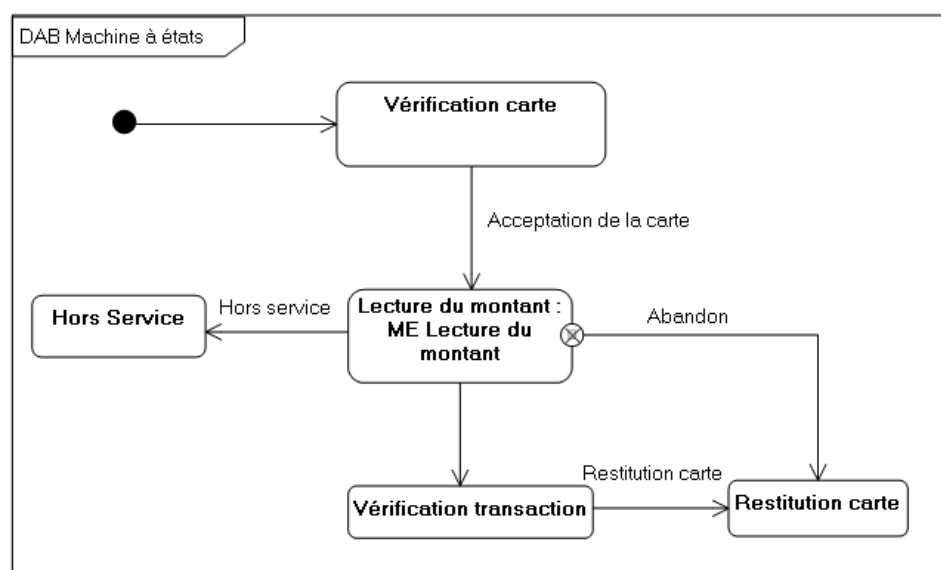
PRÉSENTATION DU DIAGRAMME DE MACHINE À ÉTATS

Une machine à état est l'ensemble des états et des transitions entre états qui définissent le cycle de vie d'un objet variable dans le temps.

Le diagramme de machine à état permet de représenter cet enchaînement d'états que peut prendre un objet en réponse aux interactions avec les objets (internes ou externes au système étudié) qui peuplent son environnement.

Exemple de diagramme de machine à d'états

Le diagramme suivant décrit les comportements possibles d'un distributeur automatique.



Créer un diagramme de machine à états

Un diagramme de machine à états se crée depuis une machine à état.

Vous pouvez créer une machine à état depuis un paquetage, une classe ou un composant.

Pour créer un diagramme de machine à états :

1. Cliquez avec le bouton droit sur une machine à état.
2. Sélectionnez **Nouveau > Diagramme d'états**.

Le diagramme est initialisé par la création d'une région. Une région est une partie d'un état composite ou d'une machine à états qui contient des états et des transitions et dont l'exécution est autonome.

LES ÉTATS


Un état d'objet est une condition ou une situation au cours de la vie d'un objet durant laquelle il satisfait à certaines conditions, exerce une certaine activité ou attend un événement. Un état d'objet représente un intervalle de temps dont les bornes sont deux événements. Un état d'objet est une phase par laquelle passe l'objet au cours de son cycle de vie.

Exemples d'état d'objet :

- Une personne peut être :
 - Célibataire
 - Mariée
 - Divorcée
- Un article peut être :
 - Disponible
 - En stock
 - En alerte
 - En rupture de stock.
 - Etc.





Créer un état

Pour créer un état :

1. Cliquez sur la flèche noire associée au bouton **Etat**  de la barre d'insertion du diagramme.
2. Sélectionnez un type d'état.
3. Cliquez sur le plan de travail.
La fenêtre **Ajout d'un état** s'ouvre.
4. Indiquez le **Nom** de l'état et cliquez sur **Créer**.
L'état apparaît dans le diagramme.

Les types d'état

Il est nécessaire de préciser le type de l'état lors de sa création. Ce peut être :

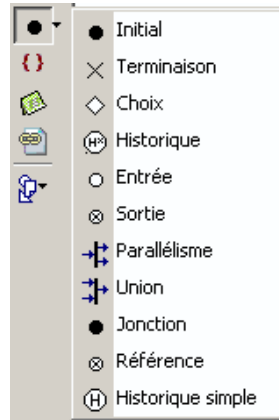
-  Un état normal : ne possède pas de sous-structure.
-  Un état composite : se compose de plusieurs états, décrits dans le diagramme.
-  Une sous-machine à état : appelle la description d'une machine à état décrite par ailleurs. Voir "[Précision comportementale d'un état](#)", page 115.
-  Un état final

Lorsque vous posez un état dans un autre, il est automatiquement relié comme composant de cet état.

Pseudo-états

Les pseudo-états sont utilisés pour spécifier des chemins complexes en combinant plusieurs transitions entre états.

Ils peuvent être de différents types : initial, terminaison, choix, historique (deep history), historique simple (shallow history), point d'entrée, point de sortie, parallélisme (fork), union (join), jonction ou référence.



Initial

Le pseudo-état initial a une seule transition en sortie vers l'état Initial de l'objet lors de sa création.

Historique

Un pseudo-état Historique représente la dernière configuration active de l'état composite qui le contient ; c'est-à-dire, la configuration active quand l'état composite a été quitté pour la dernière fois.

Historique simple

Un pseudo-état historique simple représente le plus récent sous-état actif d'un état composite (sans les sous-états de ce sous-état).

Parallélisme

Un parallélisme (fork) sépare une transition en plusieurs transitions concurrentes.

Union

Une union (join) est le regroupement de plusieurs transitions en une seule.

Choix

Représente le choix d'une transition entre plusieurs transitions possibles.

Jonction

Une jonction est utilisée pour définir des chemins de transition complexes entre plusieurs états.

Entrée

C'est un point d'entrée d'une machine à état ou d'un état composite.

Sortie

C'est un point de sortie d'une machine à état ou d'un état composite.

Référence

C'est une référence à une entrée ou à une sortie d'une machine à état ou d'un état composite

Terminaison

L'entrée dans ce pseudo-état implique une terminaison complète de la machine à état.

Historique

Un état **Historique** représente la dernière configuration active d'un état composite ; c'est-à-dire la configuration active quand l'état composite a été quitté pour la dernière fois.

Un état **Historique simple** représente le plus récent sous-état actif de l'état composite.

Exemple :

Prenons l'état "Marié" comme dernière configuration active. Cet état a pour sous-états "Avec enfants" et "Sans enfants". Dans le cas d'un historique, le sous-état "Avec enfants" ou "Sans enfant" est précisé. Dans le cas d'un historique simple, seul l'état "Marié" est pris en compte.

Précision comportementale d'un état

Un état peut être composé de sous-états.

Pour décrire la composition d'un état dans un diagramme :

1. Ouvrez le menu contextuel d'un état et cliquez sur la commande **Nouveau > Précision comportementale**.
La fenêtre de création d'un diagramme de machine à états apparaît.
2. Cliquez sur **Créer**.
Le diagramme correspondant s'ouvre.

Vous pouvez également définir la composition d'un état en lui associant une machine à état, nouvelle ou existante :

1. Ouvrez la fenêtre de propriétés de l'état décrit.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Dans le champ **Précision comportementale**, créez une machine à état ou recherchez une machine à état existante.


Propriétés d'un état

Pour accéder aux propriétés d'un état :

1. Cliquez avec le bouton droit sur l'état.
2. Sélectionnez **Propriétés**.
La fenêtre de propriétés de l'état s'ouvre.

Elle vous permet :

- De modifier le **Nom** de l'état.
- D'indiquer si ses sous-états sont **Concurrents**, c'est-à-dire s'ils peuvent être exécutés simultanément ou non.
- D'indiquer la **Précision comportementale** (dans le cas d'un état complexe). Voir "[Précision comportementale d'un état](#)", page 115.
- De préciser les **Activités** qui peuvent être effectuées en entrée, en sortie ou pendant que l'objet est dans cet état.

 Le contenu de la fenêtre de propriétés d'un état varie en fonction du type de l'état.

LES TRANSITIONS ENTRE ÉTATS

Le passage d'un état à un autre est matérialisé par une *transition*.



Une transition est le passage d'un objet d'un état dans un autre. Une transition est une réponse d'un objet à un événement qu'il reçoit. Quand un événement se produit et que certaines conditions sont satisfaites, l'objet va effectuer certaines actions tandis qu'il est encore dans le premier état puis passer au deuxième état.

Vous devez définir toutes les transitions qui sont autorisées. Celles qui ne sont pas définies sont interdites.

Exemples de transitions :

En ce qui concerne l'état civil d'une personne, certaines transitions sont possibles :


- Elle peut passer de l'état "célibataire" à l'état "marié".
- Elle peut passer de l'état "marié" à l'état "divorcé".

D'autres ne sont pas possibles :

- Elle ne peut pas passer de l'état "célibataire" à l'état "divorcé".
- etc.

Créer une transition

Pour créer une transition entre deux états :

1. Cliquez sur le bouton **Transition (UML)**  de la barre d'insertion.
2. Cliquez sur l'état de départ et déplacez la souris jusqu'à l'état d'arrivée.
3. Relâchez le bouton : la transition est créée.

Les types de transition

Une transition peut être de type externe, interne ou locale.

Vous pouvez préciser le type de la transition dans la fenêtre de propriétés de la transition, sous l'onglet **Caractéristiques**.

Transition externe

Une transition externe est une transition qui modifie l'état actif.

Transition interne

Une transition interne à un objet permet de prendre en compte l'arrivée d'un événement qui ne provoque pas de changement d'état de l'objet, mais une action comme l'appel d'une opération ou l'émission d'un message. Par exemple, lors d'un mouvement de stock, un article peut ne pas changer d'état si la quantité restant disponible en stock est suffisante et ne passe pas le seuil d'alerte ou de rupture.

Transition locale

Une transition locale s'applique aux sous-états d'un état composite. Elle peut provoquer un changement d'état uniquement à l'intérieur de l'état composite.

Effet d'une transition

Le déclenchement d'une transition peut être accompagné d'un effet. L'effet peut être représenté par :

- Une activité
- Une collaboration
- Une interaction
- Une machine à état

Pour définir l'effet d'une transition :

1. Ouvrez la fenêtre de propriétés de la transition.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Cliquez sur la flèche située à l'extrémité du champ **Effet (Comportement)** et créez ou reliez l'objet qui définit l'effet.

Affichage des effets d'une transition

Pour modifier l'affichage des effets de la transition.

1. Dans le diagramme de machine à état, cliquez avec le bouton droit sur la transition puis cliquez sur **Formes et détails**.
2. Sélectionnez "Effet" dans l'arbre qui s'affiche.

Vous pouvez choisir d'afficher tout ou partie des effets de la transition, avec leurs caractéristiques.

Événement déclencheur d'une transition

Dans la fenêtre de propriétés d'une transition, sous l'onglet **Événement**, vous pouvez indiquer le **Type d'Événement** qui déclenche une transition.

Ce peut être :

- Un événement quelconque
- L'appel d'une opération
- Un changement de l'objet concerné par la transition
- La création d'un objet
- La destruction d'un objet
- L'envoi d'un signal
- L'envoi d'une opération
- L'émission d'un signal par l'objet
- La réception d'un signal
- La réception d'une opération
- Un *temporisateur*



Un temporisateur est un événement déterminé uniquement par le temps qui s'écoule. Ex : Le lundi, à quatre heures, etc.

Les champs affichés sous le champ **Type d'événement** varient selon le type d'événement sélectionné.

Vous pouvez sélectionner l'objet concerné par l'effet.

Dans le cas d'une opération ou d'un signal, il est possible de préciser les valeurs des paramètres transmis.

LE DIAGRAMME D'ACTIVITÉS



Un diagramme d'activités est proche du diagramme de machine à états. A la différence du diagramme de machine à états qui décrit le comportement d'un objet via l'enchaînement d'états, le diagramme d'activités décrit le comportement d'un élément en termes d'actions.

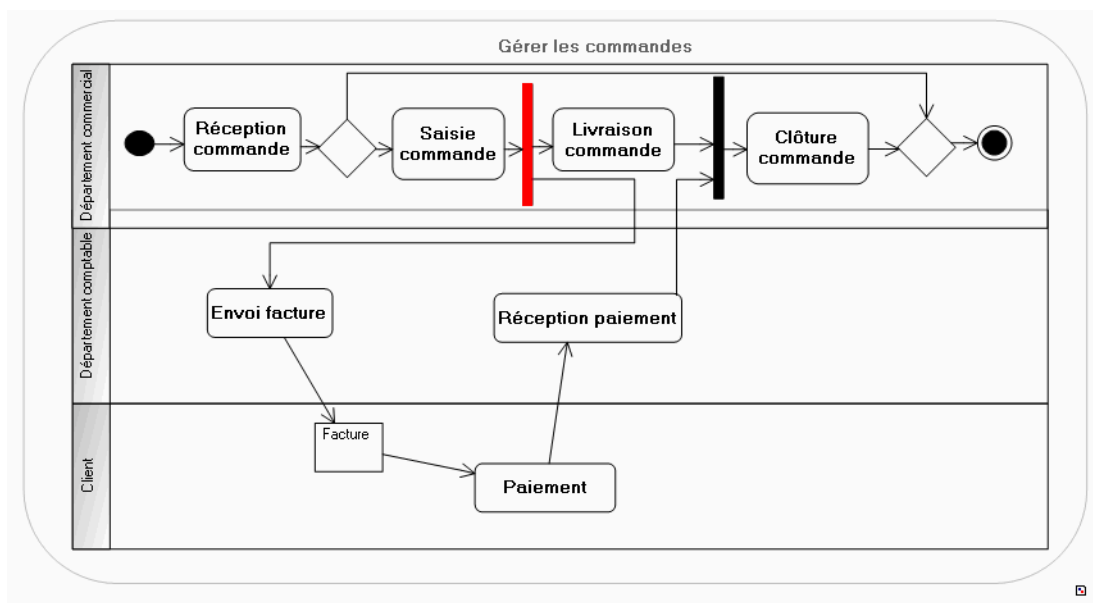
- ✓ ["Présentation du diagramme d'activités", page 122](#)
- ✓ ["Les partitions", page 123](#)
- ✓ ["Les noeuds", page 125](#)
- ✓ ["Les Flux", page 128](#)

PRÉSENTATION DU DIAGRAMME D'ACTIVITÉS

Un diagramme d'activités représente un séquençement d'étapes décrivant le comportement d'un élément du système.

Les étapes sont modélisées par des noeuds - noeuds d'action, de paramétrage ou de contrôle - coordonnés par des flux de données ou de contrôle.

Exemple de diagramme d'activités :



Créer un diagramme d'activités

Un diagramme d'activités se crée depuis une activité.

Vous pouvez créer une activité depuis un paquetage, un composant ou une classe.

Pour créer un diagramme d'activités :

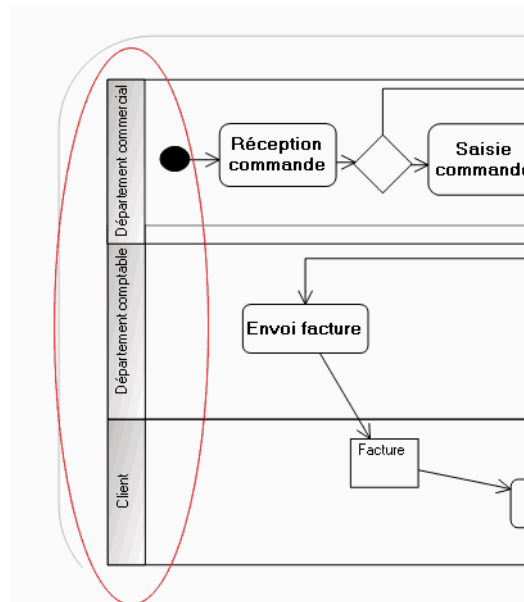
1. Cliquez avec le bouton droit sur l'activité concernée.
2. Dans le menu contextuel qui s'affiche, cliquez sur **Nouveau > Diagramme**.
3. Dans la fenêtre qui apparaît, sélectionnez "Diagramme d'activités" et cliquez sur le bouton **Créer**.
Le nouveau diagramme d'activités s'ouvre.

LES PARTITIONS

Un diagramme d'activités peut être découpé en partitions. Chaque partition contient des noeuds ou des actions ainsi que les flux entre ces éléments.


Vous pouvez utiliser des partitions pour organiser les tâches ou pour spécifier l'élément responsable de la mise en œuvre d'un ensemble de tâches.

Pour plus de détails sur les couloirs, reportez-vous à la section "Utiliser les couloirs" du guide **MEGA Common Features**.



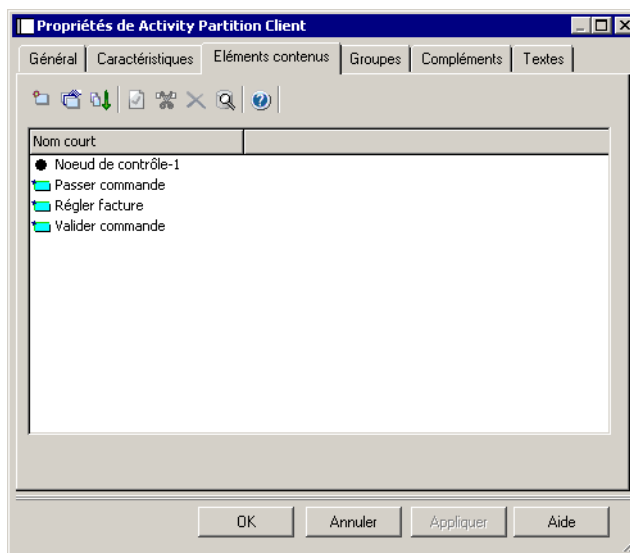
Créer une partition

Pour créer une partition dans le diagramme d'activités :

1. Cliquez sur le bouton **Partition**  de la barre d'insertion d'objets.
2. Indiquez son nom.
3. Indiquez l'élément représenté par la partition. Il s'agit de l'élément qui met en oeuvre les éléments de la partition. Ce peut être un acteur, une classe ou un composant.
4. Cliquez sur **OK**.

Propriétés d'une partition

L'onglet **Éléments contenus** de la fenêtre de propriétés de la partition présente les éléments exécutés dans la partition.



L'onglet **Compléments** permet de rattacher la partition aux contraintes qui y sont mises en oeuvre.

LES NOEUDS

Les noeuds permettent de modéliser les étapes de l'activité. Il existe trois types de noeuds dans **MEGA** :

- Les noeuds d'actions. Voir "[Les actions](#)", page 125.
- Les noeuds de paramétrage
- Les noeuds de contrôle
- Les pins d'entrée, de sortie et d'échange

Les actions

Les actions sont les étapes élémentaires du comportement représenté par l'activité.

La coordination des actions est réalisée à l'aide de flux de contrôle et de flux de données.

Créer une action

Pour créer une action :

1. Sélectionnez le bouton correspondant au type de l'action dans la barre d'insertion d'objets du diagramme, puis cliquez sur le plan de travail. La fenêtre d'ajout d'une action du type choisi s'ouvre.
2. Indiquez son nom et cliquez sur **Créer**.

Types d'action

Dans la fenêtre de propriétés de l'action, sous l'onglet **Caractéristiques**, vous pouvez préciser le type de l'action. Ce peut être :

- L'appel d'une opération d'un autre objet
- La création d'un objet
- La destruction d'un objet
- L'exécution d'une opération locale à l'objet
- L'émission d'un signal par l'objet
- La destruction finale de l'objet
- Etc.

Noeuds de paramétrage

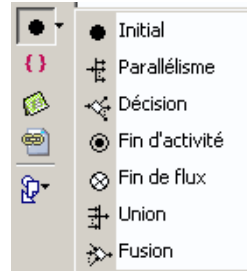
Les noeuds de paramétrage d'une activité décrivent les entrées ou les sorties de cette activité.

Ils transmettent les paramètres à l'activité par l'intermédiaire des flux qu'ils émettent ou qu'ils reçoivent.

Noeuds de contrôle

Un noeud de contrôle coordonne les flux entre les noeuds d'une activité.

Un noeud de contrôle peut être de type initial, final, décision, fusion (merge), parallélisme (fork) ou union (join).



Types de noeud de contrôle

Initial

Un noeud initial indique où débute le flux de contrôle lorsque l'activité est invoquée. Une activité peut avoir plusieurs noeuds initiaux.

Final

Lorsqu'un jeton atteint un noeud final d'activité, tous les flux de l'activité sont stoppés. Au contraire, un noeud final de flux détruit les jetons qui lui arrivent mais n'a aucun effet sur les autres jetons de l'activité.

Décision

Une décision fait le choix d'un seul flux entre plusieurs flux sortants possibles. Les flux sortants sont sélectionnés en fonction de leur condition de garde.

Fusion

Une fusion (merge) rassemble plusieurs flux alternatifs entrants en un seul flux sortant. Elle n'est pas utilisée pour synchroniser des flux concurrents mais pour accepter un seul flux parmi plusieurs.

Parallélisme

Un parallélisme (fork) sépare un flux en plusieurs flux concurrents. Les jetons arrivant à un parallélisme sont dupliqués à travers les flux sortants.

Union

Une union (join) synchronise des flux multiples. Quand tous les flux en entrée sont disponibles, le flux en sortie est déclenché.

Pins d'entrée, de sortie et d'échange

Pour spécifier les valeurs en entrée d'une action et les valeurs de retour, on utilise des nœuds d'objets appelés pins (pin en anglais) d'entrée ou de sortie. L'action ne peut débuter que si une valeur est affectée au pin d'entrée. De même, quand l'action se termine, une valeur doit être affectée au pin de sortie.

Pin d'entrée

Un pin d'entrée supporte les valeurs d'entrée qui doivent être consommées par une action et qu'il reçoit de la part d'autres actions.

Pin de sortie

Un pin de sortie supporte les valeurs de sortie qui sont produites par une action et fournit ces valeurs à d'autres actions à travers des flux.

Pin d'échange

Un pin d'échange est utilisé pour représenter les données échangées entre deux actions.

LES FLUX

Le passage d'un noeud à un autre est matérialisé par un flux.

Flux de contrôle

Un flux de contrôle démarre un noeud d'action lorsque le précédent est terminé. Les objets et les données ne peuvent pas être transmis par un flux de contrôle.

Flux d'objets

Un flux d'objets permet de transmettre des données ou objets d'un noeud à un autre à l'intérieur d'une activité.

LES DIAGRAMMES D'INTERACTION



Les diagrammes d'interaction, c'est-à-dire le diagramme de séquence, le diagramme de communication et le diagramme de vue générale d'interaction, représentent une série d'interactions entre objets, ordonnée dans le temps. Ils montrent une ou plusieurs histoires possibles du système.

Les points suivants sont abordés ici :

- ✓ ["Les interactions", page 130](#)
- ✓ ["Le diagramme de séquence", page 131](#)
- ✓ ["Le diagramme de communication", page 143](#)
- ✓ ["Le diagramme de vue générale d'interaction", page 145](#)

LES INTERACTIONS

Une interaction décrit le comportement d'un système dans un contexte particulier par les échanges de messages entre les éléments de ce système.

Quand les diagrammes de machines à état ou d'activités étudient des comportements individuels, les diagrammes d'interaction se concentrent sur la coopération d'un ensemble d'objets.

Créer une interaction

Vous pouvez créer une interaction depuis :

- Un composant
- Un paquetage
- Une classe

Créer un diagramme d'interaction

Le diagramme de séquence, le diagramme de communication et le diagramme de vue générale d'interaction se crée depuis une interaction.

Pour créer un diagramme d'interaction :

1. Cliquez avec le bouton droit sur une interaction.
2. Dans le menu contextuel qui s'affiche, cliquez sur **Nouveau > Diagramme**.
Une fenêtre contenant les différents types de diagramme possibles apparaît.
3. Sélectionnez le type de diagramme voulu et cliquez sur le bouton **Créer**.

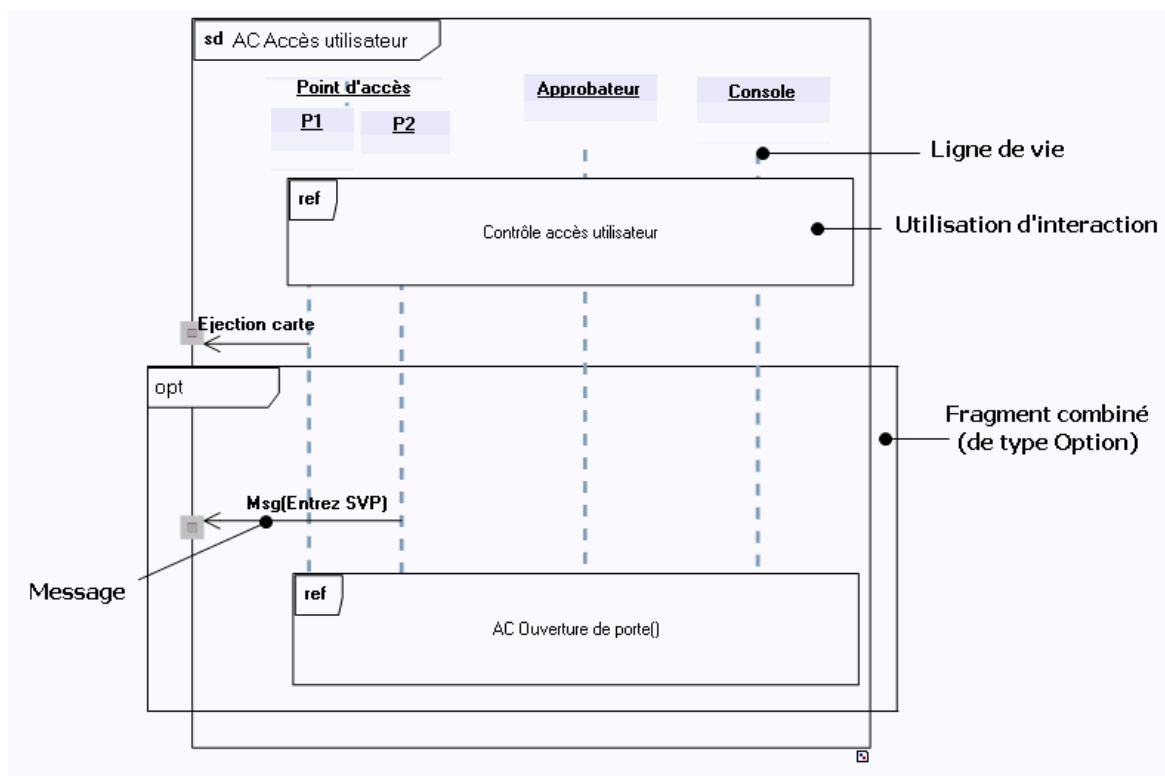
LE DIAGRAMME DE SÉQUENCE

Le diagramme de séquence met en évidence la chronologie des messages échangés entre les objets participant à une interaction. Ces objets sont représentés dans le diagramme par leurs lignes de vie.

Exemple de diagramme de séquence

Le diagramme ci-dessous décrit le comportement d'un distributeur automatique :

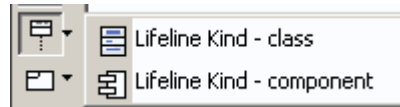
- Deux points d'entrée (représentés par des lignes de vie) donnent lieu à un contrôle d'accès de l'utilisateur. Ce contrôle est décrit dans une interaction.
- Selon le résultat du contrôle, l'accès est refusé et la carte de l'utilisateur éjectée ou l'ouverture de la porte est actionnée.
- Un comportement optionnel (représenté par un fragment combiné) peut influencer l'ouverture de la porte.



Les lignes de vie

Une ligne de vie représente un participant dans une interaction.

Les lignes de vie sont des instances de différents types (de classe, d'acteurs, etc.). La flèche située à droite du bouton de ligne de vie offre un raccourci vers les types d'objets Classe ou Composant, plus fréquemment utilisés.




Dans un diagramme de séquence, le temps est représenté comme s'écoulant du haut vers le bas le long des lignes de vie de ces objets. Entre ces objets transitent des instances de messages.


➤ Les instances représentées dans un diagramme de séquence peuvent être des instances de classe, d'acteur, de paquetage, de cas d'utilisation, de composant ou de nœud, ce qui permet de définir des diagrammes de séquence au niveau de détail souhaité.

Créer une ligne de vie

Pour créer une ligne de vie :

1. Cliquez sur le bouton **Ligne de vie** .
2. Cliquez dans le diagramme.
Une fenêtre s'ouvre.
3. Saisissez le nom de la ligne de vie.
4. Indiquez l'élément représenté par la ligne de vie.
5. Cliquez sur **Créer**.
La ligne de vie apparaît dans le diagramme.

Pour insérer plusieurs objets existants sur le diagramme en une fois :

1. Cliquez successivement sur le bouton de ligne de vie et sur le bouton Rechercher .
2. Dans la fenêtre qui apparaît, cliquez sur **chercher**.
3. En maintenant la touche <CTRL> enfoncée, sélectionnez dans la liste les objets qui vous intéressent et glissez-les dans le diagramme.

Propriétés d'une ligne de vie

Pour accéder aux propriétés d'une ligne de vie :

- Cliquez avec le bouton droit sur l'instance et sélectionnez **Propriétés**.

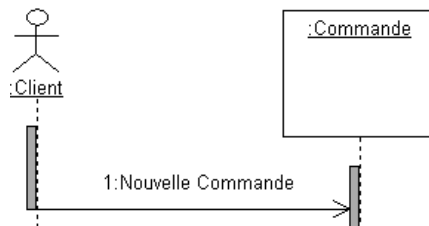
Vous pouvez sélectionner le **Type** de l'objet (Acteur, Classe, etc.), préciser de quelle **Classe**, **Acteur**, etc. cet objet est une instance et indiquer son *stéréotype*.

Les messages

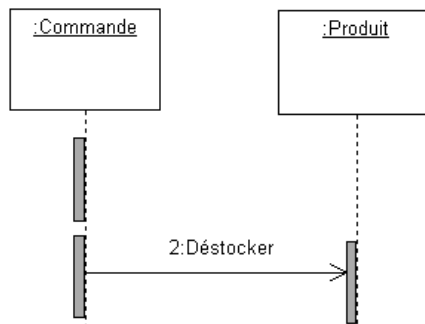
Un message définit une communication particulière entre les lignes de vie d'une interaction. Il spécifie l'émetteur et le récepteur par l'intermédiaire de spécifications d'occurrence, ainsi que le type de communication. Cette communication peut être, par exemple, l'émission d'un signal, l'invocation d'une opération, la création ou la destruction d'une instance.

Exemples de messages échangés :

1) Le message envoyé par l'acteur "Client" à la classe "Commande" transporte le signal "Nouvelle commande".



2) Le message envoyé par la classe "Commande" à la classe "Produit" appelle l'opération "Déstocker".



Créer un message

Pour créer un message dans le diagramme de séquence :

1. Cliquez sur le bouton **Message** de la barre d'insertion d'objets en sélectionnant le type de message voulu.



2. Allez de la ligne pointillée sous le premier objet à celle qui est sous le deuxième objet en maintenant le bouton gauche de la souris enfoncé. Le message échangé entre les deux objets se dessine.

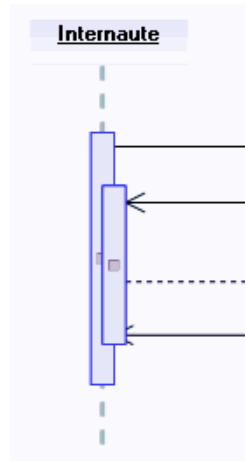
Types de messages

Vous pouvez créer quatre types de messages :

- Dans un message de type "Complet", l'émetteur et le destinataire sont tous les deux définis.
- Dans un message "Perdu", seul l'émetteur est connu. On considère ici que le message n'atteint jamais sa destination.
- Dans un message "Trouvé", seul le destinataire est connu. C'est le cas lorsque l'origine du message se situe en dehors du contexte de description.
- Dans un message de type "Inconnu", ni l'émetteur ni le destinataire ne sont définis.


Occurrence d'exécution

Une occurrence d'exécution d'une ligne de vie (execution specification) représente une unité d'action ou de comportement qui se déroule à partir d'une occurrence d'événement de début jusqu'à une occurrence d'événement de fin.



Créer une occurrence d'exécution

Pour créer une occurrence d'exécution :

1. Dans le diagramme de séquence, cliquez sur le bouton **Occurrence d'exécution**  de la barre d'insertion d'objets.
2. Positionnez-la sur la ligne de vie concernée.
L'occurrence apparaît dans le diagramme.

Occurrence d'événement

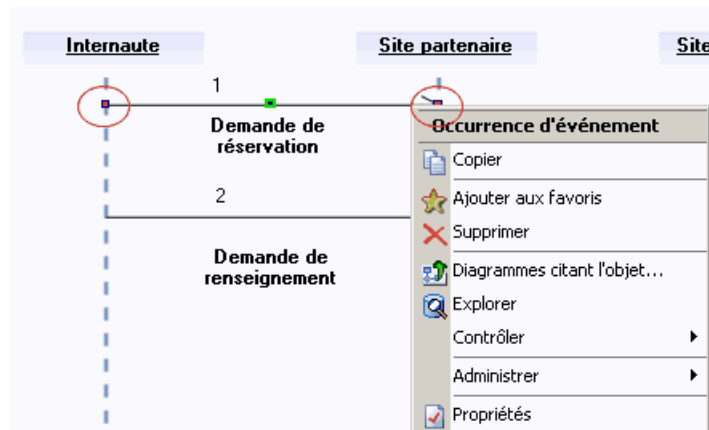
La création d'un message ou d'une occurrence d'exécution entraîne automatiquement la création d'occurrences d'événements.

Une occurrence d'événement (Occurrence Specification) est un point syntaxique à l'extrémité d'un message ou au début ou à la fin d'une occurrence d'exécution.

Les occurrences d'événement sont ordonnées le long d'une ligne de vie.

Ce sont les unités sémantiques de base d'une interaction.

Vous pouvez accéder au menu contextuel d'une occurrence d'événement en cliquant avec le bouton droit sur l'une des extrémités d'un message.

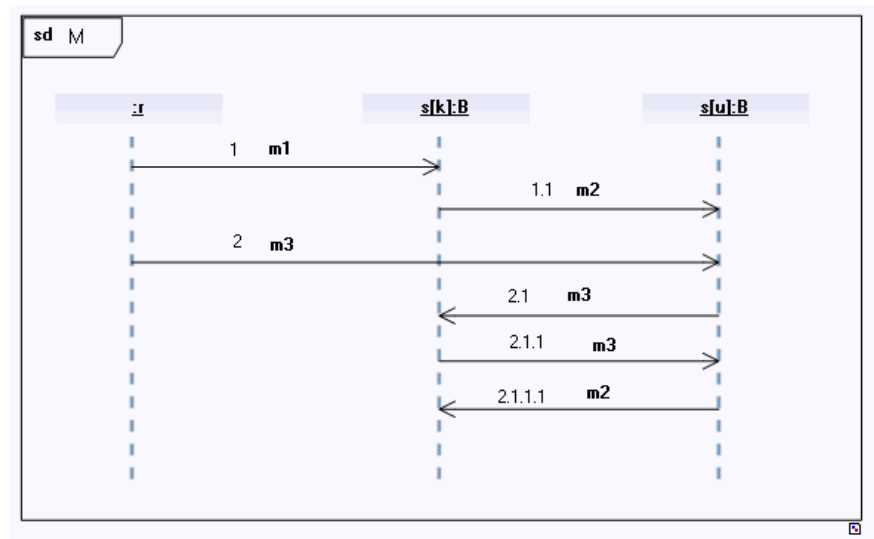


Calcul des numéros de séquence

A partir du positionnement des occurrences d'événement, un outil de calcul permet d'ordonner les messages et les occurrences d'exécution.

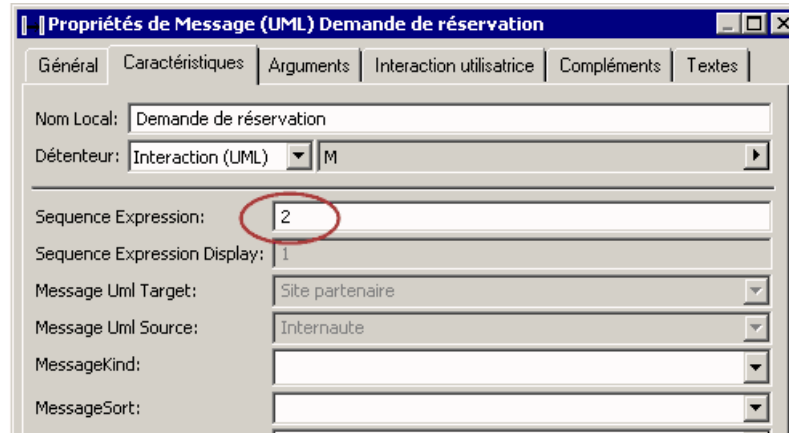
Pour ordonner les messages circulant entre des lignes de vie d'une interaction :

1. Ouvrez le menu contextuel de l'interaction décrite.
2. Cliquez sur **Calculer les numéros de séquence**.
L'outil applique automatiquement des numéros aux messages.



Vous pouvez modifier manuellement le numéro de séquence d'un message dans la fenêtre de propriétés du message :

- Cliquez sur l'onglet **Caractéristiques** et modifiez la valeur du champ **Sequence Expression**.



Lorsque vous relancez le calcul des numéros de séquence, ce dernier met à jour le séquençement en fonction des modifications apportées.

Fragment combiné

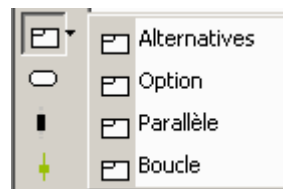
Un fragment combiné permet de décrire de manière concise plusieurs séquences d'exécution.

Un fragment combiné est défini par un opérateur d'interaction et les opérandes d'interactions correspondants.

Créer un fragment combiné

Pour créer un fragment combiné :

1. Dans la barre d'insertion d'objets du diagramme de séquence, cliquez sur le bouton **Fragment combiné**.
Vous pouvez associer au fragment combiné différents types d'opérateur d'interaction. La flèche située à droite du bouton offre un raccourci vers quatre d'entre eux. Voir ["Type d'opérateur d'interaction", page 138](#).

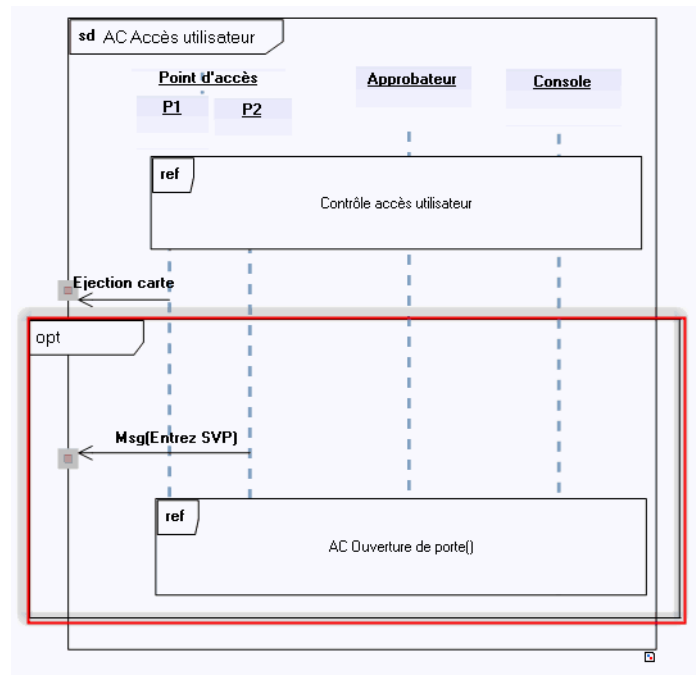


2. Cliquez dans le diagramme.
La fenêtre de création du fragment combiné apparaît.

3. Indiquez son **Nom** et le **Type d'opérateur d'interaction** si ce n'est pas déjà fait.
4. Cliquez sur **Terminer**.

Un fragment combiné est représenté par un rectangle dont l'angle supérieur gauche affiche le type d'opérateur d'interaction.

Dans l'exemple ci-dessous, un fragment combiné de type optionnel traduit un comportement susceptible de contrarier le séquençement normal (l'ouverture de la porte).



Type d'opérateur d'interaction

Le type d'opérateur d'interaction conditionne la signification du fragment combiné. Il existe différents types d'opérateurs : seq, alt, opt, break, par, strict, loop, region, neg, assert, ignore et consider.

Alternatives

Alt exprime la possibilité de choisir entre différents comportements possibles en évaluant les conditions de garde associées à chacun des opérandes. Au plus un des opérandes pourra être exécuté.

L'opérande gardé par Else est choisi lorsqu'aucune des autres conditions n'est réalisée.

Option

Opt représente un choix entre l'unique opérande proposé ou aucun.

Arrêt (Break)

Break représente un scénario d'arrêt qui est exécuté à la place du reste du fragment d'interaction englobant.

Parallèle

Par implique que les différents opérandes peuvent être exécutés en parallèle. Les occurrences des événements des divers opérandes d'interaction peuvent être entrelacées de toutes les façons tant que l'ordre imposé par chaque opérande est préservé.

Séquence faible (Weak Sequencing)

Seq désigne un entrelacement faible entre les comportements des opérandes défini par trois propriétés :

- L'ordre des occurrences d'événements à l'intérieur de chacun des opérandes est maintenu dans le résultat.
- Les occurrences d'événements de différentes lignes de vie venant de différents opérandes peuvent apparaître dans n'importe quel ordre.
- Les occurrences d'événements d'une même ligne de vie venant de différents opérandes sont ordonnés de telle sorte que l'occurrence d'événement du premier opérande apparaît avant celle du deuxième.

Séquence stricte (Strict Sequencing)

Strict définit un séquençement strict entre les comportements des opérandes.

Négation (Negative)

Neg représente un opérande invalide.

Région critique

Critical représente une région qui doit être traitée de manière atomique, ce qui signifie que des occurrences d'événement ne peuvent pas être entrelacées avec celles de la région critique.

Ignorer / Considérer

Consider et Ignore nécessitent qu'une liste de messages pertinents soient spécifiés.

Ignorer indique que les types de certains messages sont ignorés dans le fragment combiné.

Consider signifie que seuls certains messages vont être considérés à l'intérieur du fragment combiné. C'est équivalent à définir tous les autres messages comme 'ignorés'.

Assertion

Assert représente une séquence qui est la seule valide pour un message donné.

Ainsi, toute séquence définie par un fragment d'interaction qui commence par les messages qui aboutissent à la séquence définie par le bloc assert et qui continue par un échange de messages ne respectant pas le bloc assert doit être définie comme invalide.

Les assertions sont fréquemment utilisées en combinaison avec les types Ignore et Consider.

Boucle

Loop permet d'indiquer que l'opérande d'interaction sera répété un certain nombre de fois. Il est possible de spécifier un nombre minimum et un nombre maximum de boucles, ainsi qu'une expression de continuation de la boucle.

Opérande d'interaction

Un opérande d'interaction est contenu dans un fragment combiné et représente un opérande de l'expression donnée par le fragment combiné englobant. Il peut être conditionné par une contrainte d'interaction qui sert de condition de garde.

Créer un opérande d'interaction

Pour créer un opérande d'interaction :

1. Faites un clic droit sur le fragment combiné qui contient l'opérande d'interaction.
2. Sélectionnez **Nouveau** > **Opérande d'interaction**.
3. Nommez l'opérande et cliquez sur **OK**.

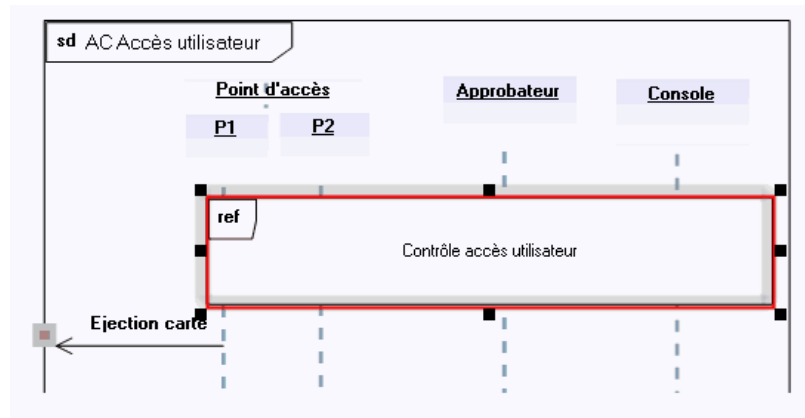
Créer une contrainte d'interaction

Pour créer la contrainte d'interaction qui va conditionner l'opérande :


1. Ouvrez la fenêtre de propriétés de l'opérande d'interaction.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Dans le cadre **Condition**, cliquez sur **Nouveau**.
4. La condition est représentée par une contrainte. Définissez la contrainte et cliquez sur **OK**.

Utilisation d'interaction

Une utilisation d'interaction se réfère à une interaction. C'est un moyen de copier le contenu de l'interaction référencée à l'endroit de l'occurrence d'interaction.



Pour créer une utilisation d'interaction :

1. Cliquez sur le bouton **Utilisation d'interaction** .
2. Cliquez dans le diagramme.
3. Dans la fenêtre qui apparaît, indiquez son nom et l'interaction appelée.
4. Cliquez sur **Terminer**.

Vous pouvez préciser les arguments d'une utilisation d'interaction. Un argument est une valeur spécifique correspondant à un paramètre de l'interaction appelée. Aussi, une fois l'argument créé sur l'utilisation d'interaction, vous devez le mettre en correspondance avec le paramètre de l'interaction appelée.

Pour créer un argument :

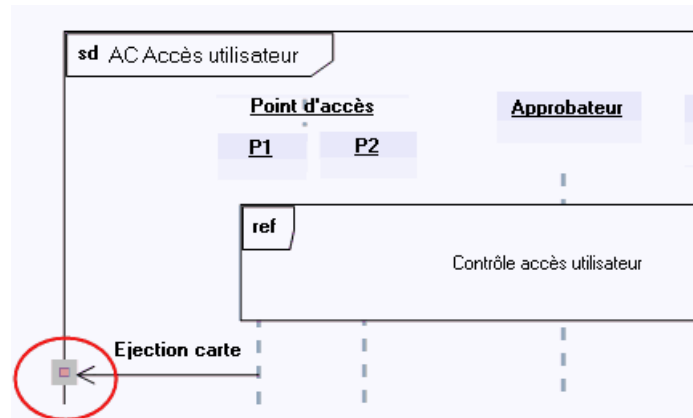
1. Ouvrez la fenêtre de propriétés de l'utilisation d'interaction.
2. Cliquez sur l'onglet **Caractéristiques**.
3. Dans le cadre **Arguments**, cliquez sur le bouton **Nouveau**.
Une spécification de valeur est créée.
Vous pouvez la renommer et préciser ses caractéristiques en ouvrant sa fenêtre de propriétés.

Pour mettre l'argument en correspondance avec le paramètre de l'interaction appelée :


1. Dans la fenêtre de propriétés de l'utilisation d'interaction, cliquez sur l'onglet **Caractéristiques**.
2. Cliquez sur la flèche située à l'extrémité du champ **Interaction called** et sélectionnez **Modifier**.
Une fenêtre affiche les paramètres de l'interaction appelée.
3. Pour chaque paramètre, cliquez dans la colonne valeur et sélectionnez la spécification de valeur qui lui correspond.

Porte

Une porte (gate) est un point de connexion entre un message extérieur à un fragment d'interaction et un message appartenant à ce fragment d'interaction.



Pour créer une porte dans le diagramme de séquence :

1. Cliquez sur le bouton **Porte**  de la barre d'insertion d'objets.
2. Cliquez sur le cadre délimitant l'interaction, là où vous souhaitez positionner la porte.
La porte apparaît dans le diagramme.

Continuation

Une continuation est un moyen syntaxique de définir le prolongement des séquences de différentes branches de fragment combiné alternatif. Les continuations sont similaires à des labels représentant des points intermédiaires dans un flot de contrôle.

LE DIAGRAMME DE COMMUNICATION

Le diagramme de communication est une représentation simplifiée d'un diagramme de séquence ; il se concentre sur les échanges de messages entre les objets au sein d'une interaction.

Le diagramme de séquence et le diagramme de communication sont isomorphes. Lorsqu'un diagramme de communication porte sur une interaction déjà décrite dans un diagramme de séquence, il est automatiquement initialisé à partir des informations contenues dans le diagramme de séquence.

Exemple

Diagramme de séquence

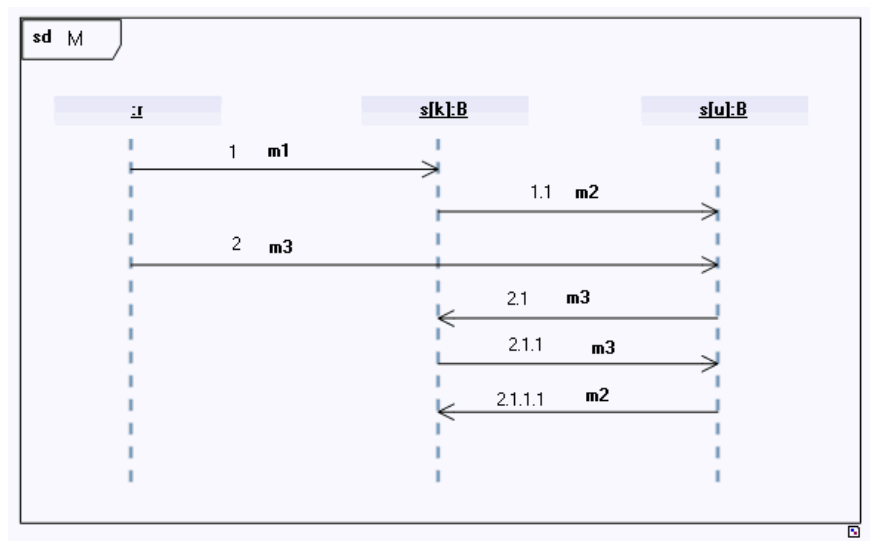
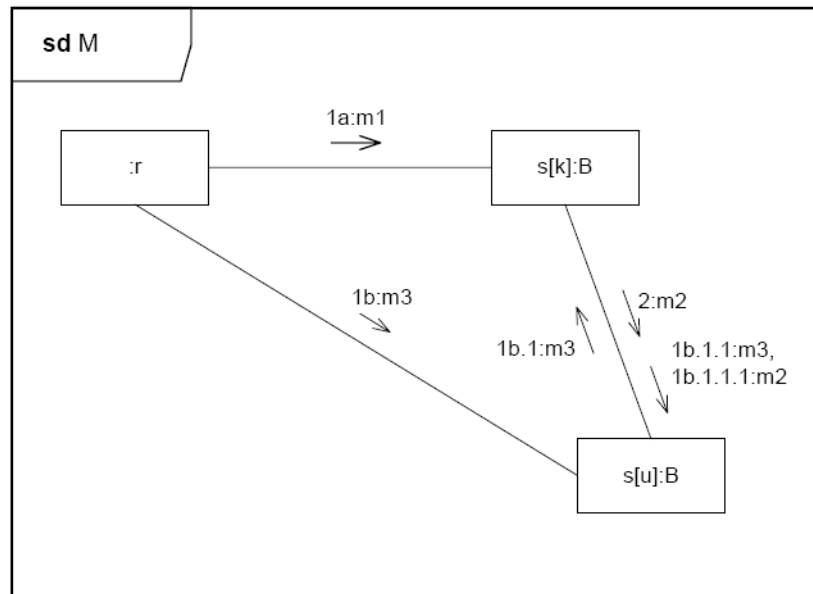



Diagramme de communication



Objets du diagramme

Les objets du diagramme de communication sont les lignes de vie et les messages transmis par des connecteurs.

Lorsque vous reliez deux lignes de vie par un connecteur , la boîte de création du connecteur propose les messages susceptibles d'être transmis.

Une fois le connecteur créé, vous pouvez également lui associer de nouveaux messages via sa fenêtre de propriétés, sous l'onglet **Message**.

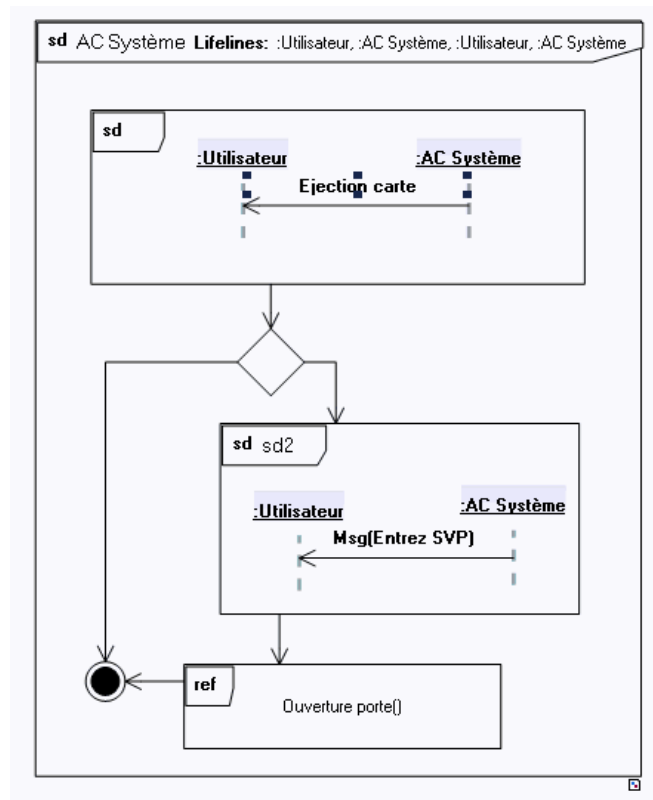


La séquence des messages est donnée par un numéro de séquence associé aux messages. Voir "[Calcul des numéros de séquence](#)", page 136.

Pour plus de détails sur les connecteurs, voir "[Les connecteurs](#)", page 103.

LE DIAGRAMME DE VUE GÉNÉRALE D'INTERACTION

Le diagramme de vue générale d'interaction permet de décrire les enchaînements possibles entre les scénarios préalablement identifiés sous forme de diagrammes de séquences. Il vise à fournir une vue d'ensemble du flux de contrôle.



Les objets représentés dans le diagramme de vue générale d'interaction sont les interactions et utilisations d'interaction, les lignes de vie, les messages, les noeuds de contrôle et les flux de contrôle.

LE DIAGRAMME DE DÉPLOIEMENT



Le diagramme de déploiement complète le diagramme de composants en mettant en avant les ressources matérielles sur lesquelles s'exécutent les composants.

- ✓ ["Présentation du diagramme de déploiement", page 148.](#)

PRÉSENTATION DU DIAGRAMME DE DÉPLOIEMENT

Le diagramme de déploiement complète le diagramme de composants. Il décrit les ressources matérielles (ordinateur, routeur etc.) qui composent le système et montre la répartition des composants sur ces matériels.

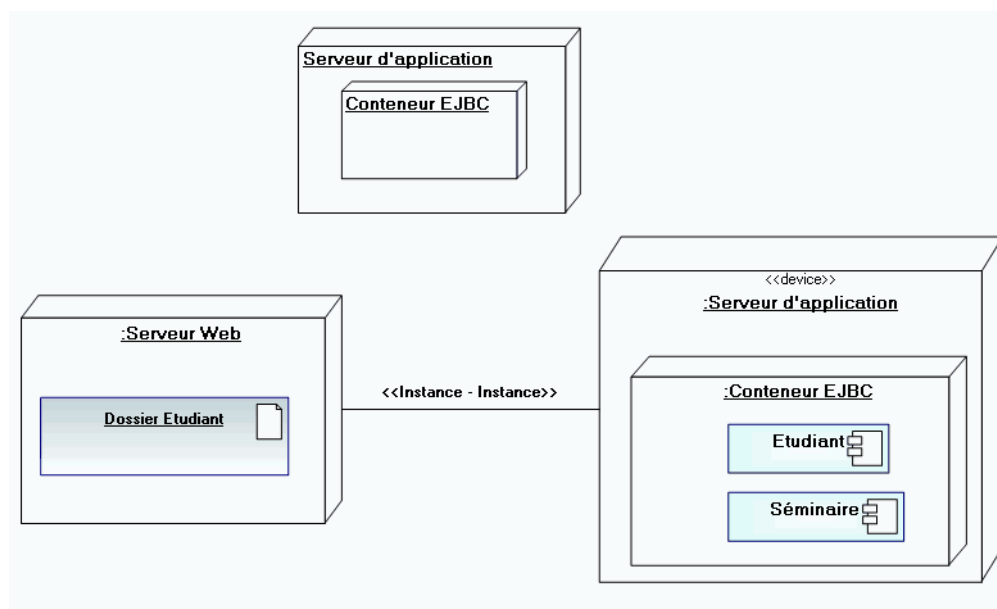
Il décrit également les connexions entre les composants ou les nœuds.

Ce diagramme permet également de préciser les interfaces requises et implémentées pour l'enchaînement des composants.

Il peut être illustré et complété par l'ajout d'instances de nœud, de composant ou de classe.

Vous pouvez créer un diagramme de composants depuis un paquetage.

Exemple de diagramme de déploiement



Objets du diagramme de déploiement


Noeud

Un noeud est un objet physique représentant une ressource informatique disposant généralement d'une mémoire et souvent de capacités de calcul et sur lesquels des composants peuvent être déployés.

Les noeuds peuvent se composer d'autre noeuds ou d'artefacts. Pour montrer qu'un composant est affecté à un nœud, il faut soit placer le composant dans le nœud, soit relier le composant au nœud par une relation de dépendance.

Voir "[Les liens de dépendance](#)", page 108.

Vous pouvez créer un noeud dans le diagramme de déploiement à l'aide du bouton

Noeud (UML)  de la barre d'insertion d'objets.

Chemin de communication

Les connexions entre noeuds sont représentées par des chemins de communication par lesquels sont échangés des signaux et des messages.


Composant

Un composant représente une partie modulaire d'un système qui encapsule son contenu et qui est remplaçable dans son environnement. Un composant définit son comportement par les interfaces qu'il fournit et celles qu'il requiert.


Un composant peut être remplacé par un autre si leurs interfaces sont conformes.

Un composant peut être un logiciel, un programme, un élément de code, etc.

Artefact

Un artefact  représente un élément d'information physique qui est utilisé ou produit par le processus de développement d'un logiciel, ou par le déploiement ou la mise en oeuvre d'un système. Ex: fichiers sources, scripts, fichiers binaires exécutables, les livrables issus d'un développement, un document produit par un traitement de texte, un message électronique, etc.

Manifestation


Une manifestation  est la restitution physique concrète dans un artefact d'un ou de plusieurs éléments de modélisation tels que des composants ou des classes.

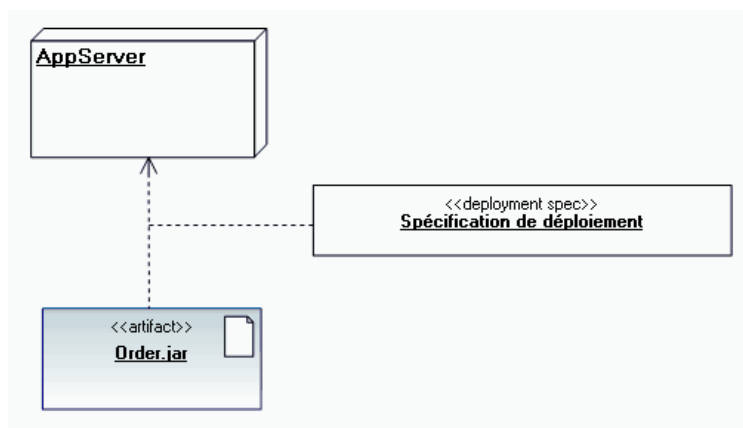
Une dépendance de manifestation a pour source un artefact et pour cible un composant ou une classe.

Spécification de déploiement

La spécification d'un déploiement permet d'indiquer l'ensemble des caractéristiques qui déterminent les paramètres d'exécution d'un artefact ou d'un composant déployé sur un noeud.

Configuration

Le bouton Configuration  permet de créer le lien entre une spécification de déploiement et un déploiement.



GLOSSAIRE



| | |
|--------------------|---|
| acteur | Un acteur représente le rôle joué par quelque chose ou quelqu'un se trouvant dans l'environnement de l'entreprise ou du système étudié. Il est en relation avec le métier de l'entreprise et interagit avec le système dans différents cas d'utilisation. Ce peut être un élément de la structure de l'entreprise tel qu'une direction, un service ou un poste de travail. |
| activité | Une activité définit une séquence d'actions élémentaires ou complexes pilotée par des paramètres. La coordination des actions est réalisée à l'aide de flux de contrôle et de flux de données. Certaines actions invoquent d'autres activités directement ou par l'intermédiaire d'une opération. Les activités peuvent être utilisées pour décrire des calculs procéduraux. Dans ce contexte, elles sont les méthodes correspondant aux opérations des classes. Les activités peuvent également être utilisées dans la modélisation du système d'information pour spécifier des processus de niveau système. |
| agrégation | L'agrégation est une forme particulière d'association qui indique que l'une des classes contient l'autre. |
| ami | Les amis d'une classe sont des classes autorisées à connaître les internes de cette classe. Il est possible de préciser les amis d'une classe dans l'onglet Compléments de la fenêtre de propriétés de la classe. |
| association | Une association est une relation existant entre deux classes. |
| argument | Un argument est une valeur spécifique correspondant à un paramètre. |

| | |
|-----------------------------|---|
| artefact | Un artefact représente un élément d'information physique qui est utilisé ou produit par le processus de développement d'un logiciel, ou par le déploiement ou la mise en oeuvre d'un système. Ex: fichiers sources, scripts, fichiers binaires exécutables, les livrables issus d'un développement, un document produit par un traitement de texte, un message électronique, etc. |
| attribut | Un attribut est une propriété nommée d'une classe. |
| cardinalité | La cardinalité est le nombre d'éléments contenus dans un ensemble. |
| cas d'utilisation | Un cas d'utilisation est une suite d'actions qui amène un résultat observable pour un acteur particulier. Des scénarios illustrent les cas d'utilisation par l'exemple. |
| classe | Une classe est un regroupement d'objets possédant des caractéristiques communes et un comportement semblable. Les classes persistantes sont des éléments de gestion considérés d'intérêt pour représenter l'activité de l'entreprise et donc conservés à cet effet. |
| classe d'association | Une classe d'association est une association qui possède aussi les propriétés d'une classe comme des attributs. |
| collaboration | Une collaboration (UML) décrit une structure collaborative entre plusieurs éléments (rôles), qui accomplissent chacun une fonction spécialisée et qui réalisent collectivement une fonctionnalité attendue du système. Son objectif est de montrer comment un système fonctionne indépendamment d'une utilisation spécifique. On en retirera donc généralement l'identité précise des classes ou des instances qui y participent. |
| composant | <p>Un composant représente une partie modulaire d'un système qui encapsule son contenu et qui est remplaçable dans son environnement. Un composant définit son comportement par les interfaces qu'il fournit et celles qu'il requiert.</p> <p>Un composant peut être remplacé par un autre si leurs interfaces sont conformes.</p> <p>Un composant peut être un logiciel, un programme, un élément de code, etc.</p> |
| composition | La composition est une agrégation forte pour laquelle la durée de vie des composants coïncide avec celle du composé. Une composition est une agrégation immuable avec une multiplicité 1. |
| condition | Une condition précise l'état dans lequel un objet doit se trouver pour qu'une action puisse être effectuée. |

| | |
|-----------------------|--|
| connecteur | <p>Un connecteur est un lien qui permet d'établir une communication entre plusieurs objets.</p> <p>Un connecteur de délégation relie le contrat externe de l'objet (tel qu'il est spécifié par ses ports et/ou ses interfaces) aux objets internes qui vont le réaliser.</p> <p>Un connecteur d'assemblage entre plusieurs objets (ou leurs ports) permet de spécifier comment un des objets fournit l'interface requise par un autre.</p> |
| contrainte | <p>Une contrainte représente un contrôle ou une règle de gestion qui doit être appliquée lors de l'exécution d'un traitement.</p> |
| dépendance | <p>Une dépendance précise que l'implémentation ou le fonctionnement d'un ou de plusieurs éléments nécessite la présence d'un ou de plusieurs autres éléments. Il existe plusieurs stéréotypes de dépendance.</p> |
| discriminant | <p>Le discriminant est l'attribut d'une généralisation dont la valeur permet de répartir les objets entre les sous-classes associées à la généralisation.</p> |
| entité | <p>L'entité est un stéréotype qui permet de caractériser des classes passives qui ne génèrent aucune interaction par elles-mêmes. Elles peuvent participer à plusieurs cas d'utilisation et survivent généralement à une interaction unique. Elles représentent des objets partagés entre les différents acteurs qui les manipulent.</p> |
| événement | <p>Un événement est un fait significatif localisé dans le temps et dans l'espace. Un événement peut avoir des paramètres. Dans les diagrammes d'états, un événement peut déclencher une transition. Ce peut être la réception d'un message, la pression sur un bouton, le départ d'un train, le 31 décembre, etc.</p> |
| état d'objet | <p>Un état d'objet est une condition ou une situation au cours de la vie d'un objet durant laquelle il satisfait à certaines conditions, exerce une certaine activité ou attend un événement. Un état d'objet représente un intervalle de temps dont les bornes sont deux événements. Un état d'objet est une phase par laquelle passe l'objet au cours de son cycle de vie.</p> |
| extension | <p>Lorsqu'un cas d'utilisation comprend trop d'alternatives et d'exceptions, ces dernières sont représentées à part dans des extensions du cas d'utilisation standard.</p> |
| généralisation | <p>Une généralisation représente une relation d'héritage entre une classe générale et une classe plus spécifique. La classe spécifique est cohérente avec la classe plus générale et en hérite ses caractéristiques et son comportement. Elle comporte cependant des informations supplémentaires. Toute instance de la classe spécifique est aussi une instance de la classe générale.</p> |

| | |
|---------------------|---|
| interface | Une interface représente la partie visible d'une classe ou d'un paquetage dans une relation contractuelle de type client - fournisseur. L'interface est un stéréotype de classe. |
| lien | Un lien entre objets représente une instance d'Association entre deux objets. |
| message | Un message décrit un flux de contrôle synchrone ou asynchrone d'un objet émetteur vers un ou plusieurs objets récepteurs. Il peut éventuellement être porteur d'information. La réception d'un message est généralement un événement générateur d'une activité chez l'objet récepteur. |
| métamodèle | Le métamodèle définit la structure du langage utilisé dans les modèles. |
| modifiable | La caractéristique modifiable permet de préciser si le rôle joué par une classe dans une association est modifiable après qu'il a été créé ou non. Par défaut, le rôle d'une classe dans une association est considéré comme modifiable. |
| multiplicité | La multiplicité précise l'intervalle entre les valeurs minimum et maximum des cardinalités possibles pour un ensemble. On l'indique en particulier pour chacun des rôles que jouent les classes dans une association. Elle peut prendre les valeurs *, 0..1, 1, 1..*, 2..*, 4..10, etc. La valeur proposée par défaut est *. |
| navigabilité | La navigabilité précise le sens dans lequel l'association entre deux classes peut être parcourue. Pour ne pas encombrer le dessin, on n'indiquera la navigabilité que lorsqu'elle n'a lieu que dans un seul sens. |
| Noeud | Un noeud est un objet physique représentant une ressource informatique disposant généralement d'une mémoire et souvent de capacités de calcul et sur lesquels des composants peuvent être déployés. |
| objet | Un objet est une entité avec une identité et des frontières clairement définies dont l'état et le comportement sont encapsulés. Son état est défini par les valeurs de ses attributs et de ses liens avec d'autres objets. Son comportement est représenté par ses opérations et ses méthodes. Un objet est une instance de classe. |
| opération | Une opération est un service qui peut être demandé à un objet pour mettre en œuvre un comportement défini. Une opération possède une signature qui permet de préciser les paramètres qui lui sont nécessaires. |

| | |
|--------------------------|---|
| paquetage | Un paquetage partitionne le domaine d'étude et les travaux associés. Il permet de regrouper divers éléments, en particulier des cas d'utilisations et des classes. Un paquetage peut aussi contenir d'autres paquetages. Les paquetages sont liés entre eux à travers des rapports contractuels définissant leur interface. |
| paramètre | Un paramètre est la spécification d'une variable qui peut être modifiée, transmise ou retournée. Un paramètre peut préciser un nom, un type et une direction. Les paramètres sont utilisés pour les opérations, les messages et les événements. |
| participation | Une participation indique qu'un acteur joue un rôle dans un cas d'utilisation. |
| persistance | La persistance précise si les objets de la classe doivent être conservés dans le temps ou s'ils ne vivent que le temps du traitement en mémoire dans l'ordinateur. |
| point d'extension | Un point d'extension identifie un point dans le comportement d'un cas d'utilisation où ce comportement peut être étendu par un autre cas d'utilisation. |
| qualificatif | Un qualificatif est un attribut dont les valeurs partitionnent l'ensemble des objets reliés à un objet à travers une association. |
| région | Une région est une partie d'un état composite ou d'une machine à états qui contient des états et des transitions et dont l'exécution est autonome. |
| rôle | Un rôle représente le comportement spécifique d'une entité participante à un contexte particulier. Un rôle peut être statique (rôle d'association) ou dynamique (rôle de collaboration). |
| signal | Un signal est un événement qui peut être invoqué explicitement. Un signal peut posséder des paramètres. Un signal peut être envoyé à un objet ou à un ensemble d'objets. Un signal peut être invoqué dans le cadre de la participation d'un acteur à un cas d'utilisation. |
| stéréotype | Un stéréotype est un type d'élément de modélisation qui permet d'étendre la sémantique du métamodèle. Les stéréotypes doivent être basés sur des types ou des classes existantes dont ils reprennent la structure. D'autres stéréotypes peuvent être créés par l'utilisateur. |
| temporisateur | Un temporisateur est un événement déterminé uniquement par le temps qui s'écoule. Ex : Le lundi, à quatre heures, etc. |

| | |
|---------------------------|--|
| transition | Une transition est le passage d'un objet d'un état dans un autre. Une transition est une réponse d'un objet à un événement qu'il reçoit. Quand un événement se produit et que certaines conditions sont satisfaites, l'objet va effectuer certaines actions tandis qu'il est encore dans le premier état puis passer au deuxième état. |
| transition interne | Une transition interne à un objet permet de prendre en compte l'arrivée d'un événement qui ne provoque pas de changement d'état de l'objet, mais qui a un effet comme l'appel d'une opération ou l'émission d'un message. |
| type de données | Un type de données permet de mettre en commun des caractéristiques communes à plusieurs attributs. Les types de données sont implémentés sous forme de classe. |
| type expression | Le type expression d'une opération précise le type de la variable retournée par l'opération à la fin de son exécution. |
| utilisation | L'association d'utilisation permet de mettre en facteur dans un cas d'utilisation particulier un comportement commun à plusieurs cas d'utilisation. Ce cas d'utilisation abstrait n'a de sens que lorsqu'il est utilisé comme partie d'un autre. |

ANNEXE : TYPE DES ATTRIBUTS



Les points suivants sont abordés ici :

- ✓ ["Types élémentaires", page 158](#)
- ✓ ["Paquetages et types élémentaires", page 160](#)
- ✓ ["Définir de nouveaux types élémentaires", page 163](#)

TYPES ÉLÉMENTAIRES

Un type élémentaire permet de mettre en commun des caractéristiques communes à plusieurs attributs. Les types élémentaires sont implémentés sous forme de classe.

Définir un type élémentaire

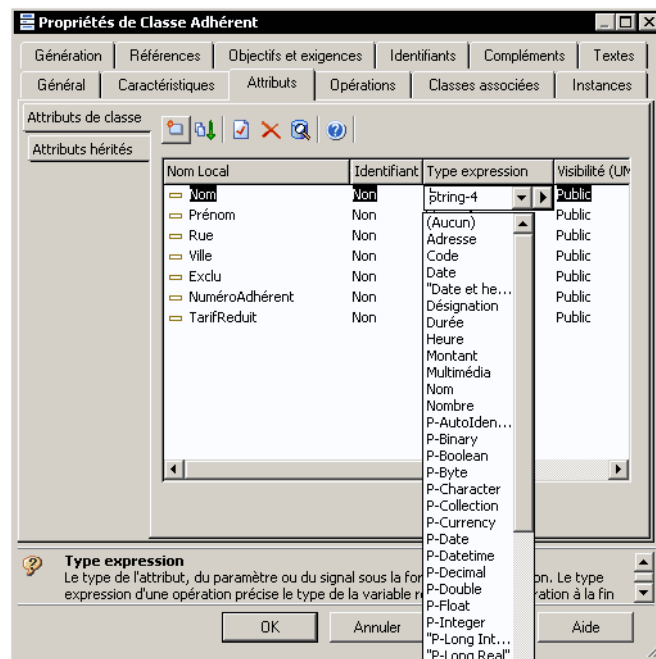
Les types élémentaires sont définis dans un diagramme de classes.

Ce sont des classes pour lesquelles on précise les points suivants :

- Ce sont des classes de stéréotype "Type élémentaire".
- Ce sont des classes "Abstraites" car elles ne sont pas destinées à être instanciées.
- Ce sont des classes "Non persistantes". Elles ne doivent en effet pas donner lieu à une table dans la base de données.

Pour renseigner le type des attributs des classes :

1. Dans la fenêtre de propriétés de la classe, cliquez sur l'onglet **Attributs**.
2. Cliquez dans le champ **Type expression** et sélectionnez le type de l'attribut à l'aide de la flèche.



Les classes proposées en standard sont :

| Types alphanumériques | | Compléments |
|-------------------------|---|--------------------|
| M-Char | Chaîne de caractères alphanumériques de taille fixe | Longueur |
| M-Varchar | Chaîne de caractères alphanumériques de taille variable | |
| Types numériques | | |
| M-Numeric | Numérique | Longueur, Décimale |
| M-Amount | Montant exprimé en monnaie | Longueur, Décimale |
| Types dates | | |
| M-Date | Date | |
| M-Time | Heure | |
| M-Datetime | Date et heure | |
| Types binaires | | |
| M-Timestamp | Identification générée automatiquement à partir de la date et de l'heure exprimée en millièmes de secondes après le 01 Janvier 1970 | |
| M-Bool | Booléen valant 0 ou 1 | |
| M-Multimedia | Chaîne binaire | |

PAQUETAGES ET TYPES ÉLÉMENTAIRES

Paquetages



Un paquetage partitionne le domaine d'étude et les travaux associés. Il permet de regrouper divers éléments, en particulier des cas d'utilisations et des classes. Un paquetage peut aussi contenir d'autres paquetages. Les paquetages sont liés entre eux à travers des rapports contractuels définissant leur interface.

L'affectation des classes à des paquetages est extrêmement structurante. En effet, comme une classe ne peut être détenue que par un seul paquetage, il est nécessaire de définir des liens clients - fournisseurs entre les paquetages de façon à ce que les paquetages qui en ont besoin puissent utiliser les classes dont ils ne sont pas détenteurs.

Ceci est particulièrement important pour les classes de type élémentaire puisqu'elles vont être utilisées pour définir les attributs des autres classes.



Règle : Une classe est détenue par un paquetage et un seul.

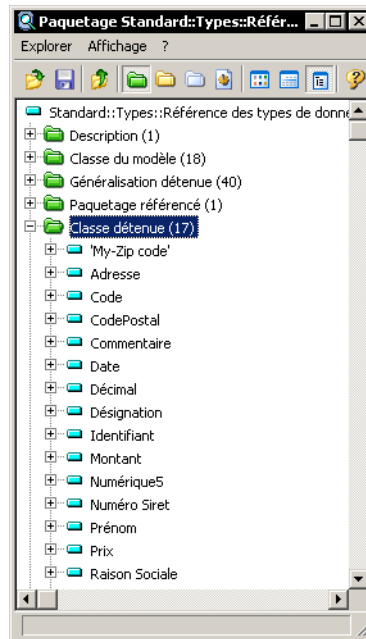
Les types élémentaires disponibles pour les attributs d'une classe dépendent du paquetage qui la détient.

Pour typer les attributs d'une classe, on proposera les types élémentaires définis pour le paquetage qui contient cette classe.

Les types élémentaires disponibles sont les classes publiques de stéréotype "Type élémentaire" détenues ou utilisées par ce paquetage ou par les paquetages dont il est client.

On peut ainsi définir un paquetage de référence (ou plusieurs) détenant les types élémentaires utilisés dans l'entreprise. Chacun des autres paquetages sera déclaré client du paquetage de référence des types élémentaires.

Dans l'exemple ci-dessous, le paquetage "Référence des types de données" détient les classes "Adresse", "Code", "Date", etc.



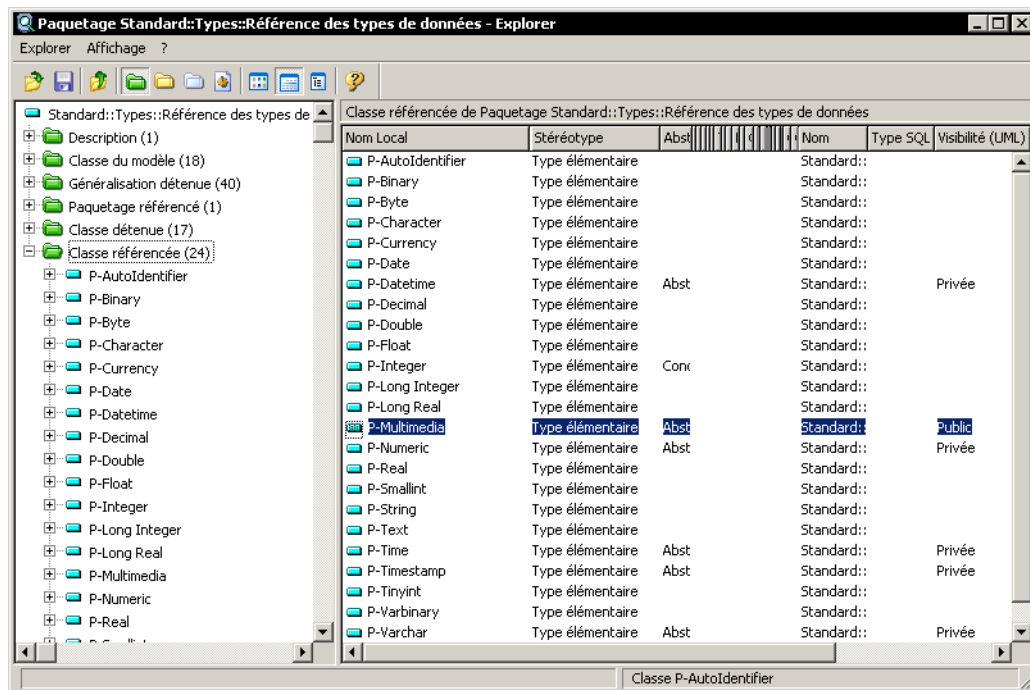
Il est référencé par les paquetages "Bibliothèque", "Gestion des commandes" etc.

Les attributs des classes de ces paquetages peuvent donc être typés à l'aide des types "Adresse", "Code", "Date", etc.

Il est également possible de préciser directement qu'un paquetage référence une classe détenue par un autre paquetage.

Dans l'exemple ci-dessous, les classes "P-Datetime", "P-Multimedia", "P-Numeric", etc., sont référencées par le paquetage "Référence des types de données" sans qu'il les détienne.

Parmi celles-ci, seule la classe "M-Multimedia" est rendue publique par ce paquetage.



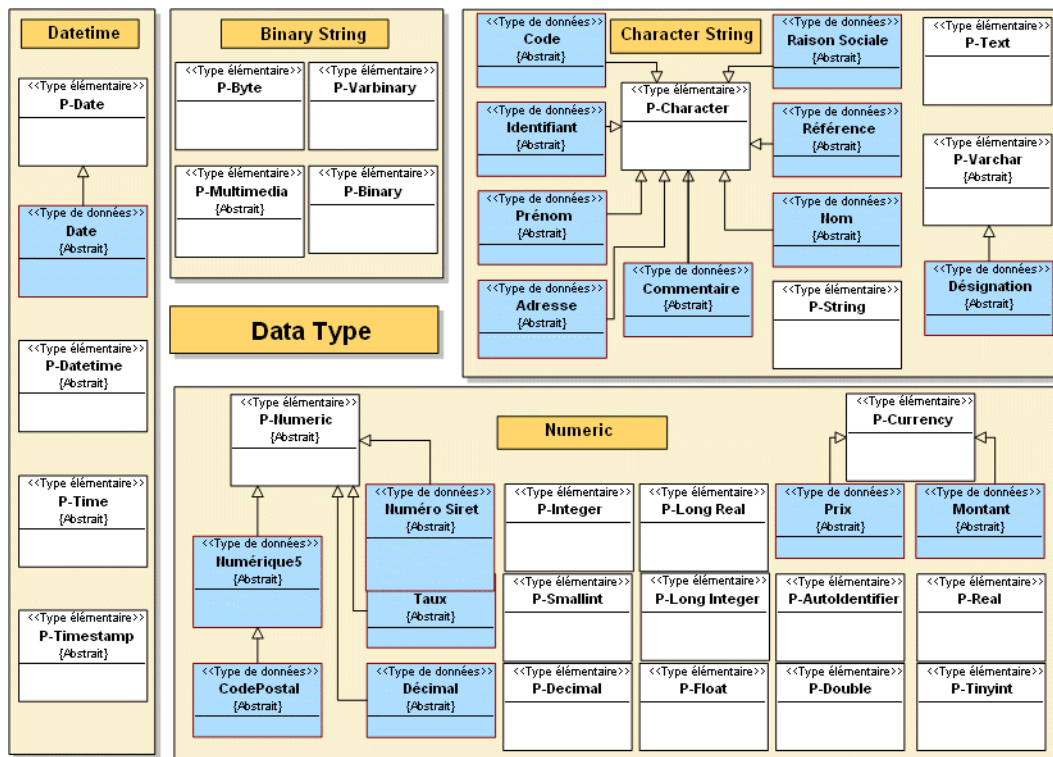
DÉFINIR DE NOUVEAUX TYPES ÉLÉMENTAIRES

De nouveaux types élémentaires peuvent être définis à l'aide d'un diagramme de classes.

Ce diagramme de classes pourra décrire, selon que l'on aura choisi ou non la structuration des classes dans des paquetages :

- Une base de données de référence.
- Le paquetage des types de référence.

Vous pouvez définir vos propres types élémentaires en les déclarant sous-classes des types élémentaires proposés en standard comme dans l'exemple ci-dessous :

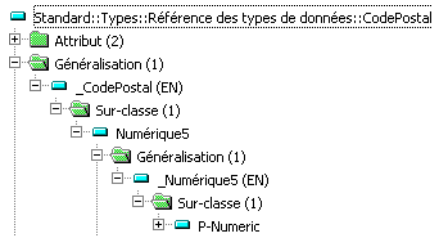


Les types élémentaires définis comme sous-classes vont hériter automatiquement des caractéristiques de leur super-classe. En particulier, la règle de transformation en datatype de la super-classe est appliquée à la sous-classe.

Il est possible de préciser sur la sous-classe une longueur et un nombre de décimales. Ceux-ci seront pris en compte pour la génération des datatypes s'ils n'ont pas déjà été définis pour la super-classe.

L'héritage peut se faire sur plusieurs niveaux.

Dans l'exemple suivant, le type élémentaire "CodePostal" est une spécialisation du type "Numérique5" de longueur 5, lui-même spécialisation du type standard "P-Numeric".

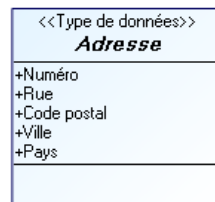


Si le nouveau type élémentaire n'est pas défini directement ou indirectement comme sous-classe d'un type élémentaire standard, il est nécessaire de mettre à jour le tableau de conversion des types élémentaires en datatypes de colonnes.

☛ Une correspondance peut également être définie directement entre un type et le datatype SQL généré pour chaque SGBD cible sans utiliser le mécanisme d'héritage (voir "Correspondances entre types pivots et datatypes" dans le guide **MEGA Database Builder**).

Type élémentaire composé

On peut définir un type élémentaire composé en lui précisant une liste d'attributs.

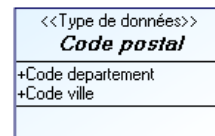
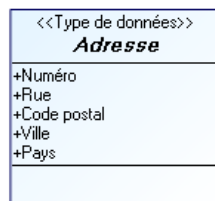


Ici le type Adresse est composé du numéro, de la rue, du code postal, de la ville et du pays.

Un attribut de type Adresse donnera lieu lors de la dérivation à ces cinq colonnes.

Il est possible de décomposer un type à plusieurs niveaux en affectant un type décomposé à l'un de ses attributs.

Par exemple, on peut décomposer le code postal en code ville et code département :



INDEX



A

- abstraite
 - classe 42
- acteur
 - stéréotype de classe 43
 - UML 22
- action
 - UML
 - créer* 125
- active
 - classe 42
- activité
 - couloir
 - UML* 123
 - définition 151
 - diagramme d'activités 121
 - état 116
- affichage
 - transition
 - UML* 118
- agrégation
 - définition 151
 - rôle 67
- ami
 - définition 151
- argument
 - définition 151
 - paramètre 60
- arranger
 - dessin 94
- arrière-plan 25
- association 61
 - classe d'association 71
 - créer 62
 - définition 151
 - réflexive 72
 - rôles 62
- ternaire 72
- attribut 46
 - affichage 57
 - calculé 50
 - définition 152
 - hérité 48
 - lecture seule, lecture seule
 - attribut* 49
 - multiplicité 47, 49
 - persistant 49
 - propriétés 49
 - statique 49
 - type 47, 49
 - type des attributs 157
 - valeur 87
 - valeur initiale 50
 - visibilité 47, 50
- auxiliaire
 - stéréotype de classe 43

B

- barycentrique
 - réorganisation 94

C

- calculé
 - attribut 50
- cardinalité
 - définition 152

| | |
|--|----------|
| cas d'utilisation | |
| créer | 23 |
| définition | 152 |
| diagramme | 23 |
| extension | 27 |
| inclusion | 27 |
| interface | 32 |
| point d'extension | 29 |
| classe | 39 |
| abstraite | 42 |
| active | 42 |
| attributs | 46 |
| caractéristiques | 42 |
| créer | 40 |
| définition | 152 |
| diagramme de paquetages | 100 |
| éditeur | 90 |
| feuille | 42 |
| généralisation | 74 |
| interface | 79 |
| opération | 51 |
| persistance | 42 |
| propriétés | 41 |
| racine | 42 |
| visibilité | 42 |
| classe d'association | 71 |
| définition | 152 |
| classe d'implémentation | |
| stéréotype de classe | 43 |
| classe meta | |
| stéréotype de classe | 43 |
| classe paramétrée | 82 |
| commentaire | |
| objet | 43 |
| composant | |
| définition | 152 |
| diagramme de composants | 101 |
| diagramme de structure composite | 106 |
| interface | 102, 106 |
| composition | 67 |
| définition | 152 |
| concurrence | |
| opération | 52 |
| concurrent | |
| état | |
| UML | 116 |
| constructeur | |
| stéréotype | 52 |
| contrainte | 84 |
| définition | 153 |
| extension | 30 |
| opération | 56 |
| contrôle | |
| signature | 54 |
| stéréotype de classe | 43 |
| corps | |
| opération | 56 |
| couloir | |
| activité | |
| UML | 123 |
| D | |
| dépendance | |
| classe | |
| <i>paquetage</i> | 100 |
| <i>paquetage</i> | 81 |
| destructeur | |
| stéréotype | 52 |
| diagramme | |
| cas d'utilisation | |
| <i>créer</i> | 20 |
| <i>ouvrir</i> | 21 |
| d'interfaces | 106 |
| réorganisation automatique | 94 |
| diagramme d'activités | 121 |
| créer | 122 |
| diagramme d'états | 112 |
| créer | 112 |
| ouvrir | 113 |
| diagramme d'interaction | 129 |
| diagramme d'objet | |
| lien | 87 |
| rôle | 88 |
| diagramme d'objets | 85 |
| diagramme de cas d'utilisation | |
| exclusion | 27 |
| généralisation | 30 |
| inclusion | 27 |
| <i>paquetage</i> | 24 |
| participation | 25 |
| diagramme de classe | |
| interface | 79 |
| diagramme de classes | 35 |
| créer | 38 |
| générer (MEGA Development) | 94 |
| diagramme de communication | |
| créer | 130 |
| présentation | 143 |
| diagramme de composants | 101 |
| diagramme de déploiement | 147 |
| diagramme de paquetages | 98 |
| créer | 98 |
| ouvrir | 99 |
| diagramme de séquence | 131 |
| créer | 130 |
| message | 133 |

| | |
|--|-----|
| diagramme de structure composite | 106 |
| diagramme de vue générale d'interaction | |
| créer | 130 |
| discriminant | |
| définition | 153 |
| généralisation | 78 |

E

| | |
|------------------------------------|-----|
| éditeur de classes | 90 |
| affichage | 90 |
| créer des objets | 92 |
| ouvrir | 90 |
| propriétés. | 91 |
| en | 89 |
| entité | |
| définition | 153 |
| stéréotype de classe | 43 |
| énumération | |
| stéréotype de classe | 44 |
| état | |
| activité. | 116 |
| historique simple. | 115 |
| précision comportementale. | 115 |
| UML | 113 |
| <i>historique</i> | 115 |
| <i>propriétés</i> | 116 |
| <i>type</i> | 113 |
| état d'objet | |
| définition | 153 |
| événement | |
| définition | 153 |
| transition transition | |
| <i>événement</i> | 118 |
| exception | |
| opération | 56 |
| expression | |
| stéréotype de classe | 44 |
| extension | |
| cas d'utilisation. | 27 |
| contrainte. | 30 |
| définition | 153 |
| externe | |
| opérateur. | 44 |

F

| | |
|-------------------------------|----|
| feuille | |
| classe. | 42 |
| focus | |
| stéréotype de classe. | 44 |
| frontière | |
| stéréotype de classe. | 44 |

G

| | |
|---|--------|
| généralisation | |
| classe. | 74 |
| créer | 31, 77 |
| définition | 153 |
| diagramme de cas d'utilisation | 30 |
| discriminant | 78 |
| réutiliser. | 31 |
| sous-classe | 76 |
| générer | |
| diagramme de classes (MEGA Development) | 94 |

H

| | |
|--------------------------|-----|
| héritage | |
| attribut. | 48 |
| multiple | 77 |
| opération | 52 |
| hiérarchique | |
| réorganisation | 94 |
| historique | |
| état | |
| UML | 115 |
| simple | |
| <i>état</i> | 115 |

I

| | |
|-----------------------------|----|
| inclusion | |
| cas d'utilisation | 27 |
| initiateur | |
| cas d'utilisation | 26 |

| | |
|--|-----|
| instance | |
| acteur | 132 |
| classe | 132 |
| créer | |
| <i>classe</i> | 132 |
| <i>objet</i> | 86 |
| diagramme de séquence | 132 |
| propriétés | |
| <i>diagramme d'interaction</i> | 132 |
| rôle | 88 |
| Interface | |
| composant | 106 |
| interface | |
| cas d'utilisation | 32 |
| composant | 102 |
| définition | 154 |
| diagramme de classe | 79 |
| interface supportée | 102 |
| requis | 102 |
| stéréotype de classe | 44 |
| interne | |
| opérateur | 44 |
| itérateur | |
| stéréotype | 52 |

L

| | |
|-----------------------------|-----|
| lancer | |
| MEGA Development | 15 |
| lecture seule | |
| rôle | 68 |
| lien | |
| association | 88 |
| définition | 154 |
| diagramme d'objet | 87 |
| rôle | 88 |
| stéréotype | 88 |
| ligne de vie | |
| UML | 123 |
| lister | |
| classe | 40 |

M

| | |
|---------------------------------|-----|
| message | |
| définition | 154 |
| diagramme de séquence | 133 |

| | |
|-------------------------|-----|
| métamodèle | |
| définition | 154 |
| méthode | |
| opération | 55 |
| modifiable | |
| définition | 154 |
| multiple | |
| héritage | 77 |
| multiplicité | |
| attribut | 47 |
| définition | 154 |
| participation | 27 |
| rôle | 63 |

N

| | |
|----------------------|-----|
| navigabilité | |
| définition | 154 |
| rôle | 66 |

O

| | |
|--------------------------------|-----|
| objet | |
| créer | 86 |
| définition | 154 |
| opérateur | |
| externe | 44 |
| interne | 44 |
| stéréotype de classe | 44 |
| opération | |
| affichage | 57 |
| classe | 51 |
| concurrence | 52 |
| consultation | 53 |
| contrainte | 56 |
| corps | 56 |
| définition | 154 |
| direction | 55 |
| exception | 56 |
| méthode | 55 |
| polymorphe | 53 |
| post-condition | 56 |
| pré-condition | 56 |
| protégée | 53 |
| séquentielle | 53 |
| signature | 53 |
| statique | 52 |
| stéréotype | 52 |

| | |
|-----------------------------|----|
| type expression. | 53 |
| UML. | 51 |
| <i>paramètres</i> | 55 |
| <i>propriétés</i> | 52 |
| valeur par défaut. | 55 |
| visibilité | 53 |
| ordre | |
| rôle | 69 |
| orthogonale | |
| réorganisation. | 94 |

P

| | |
|--|-----|
| paquetage | |
| créer | 20 |
| définition | 155 |
| diagramme de cas d'utilisation | 24 |
| diagramme de paquetages | 99 |
| référencer | 81 |
| type élémentaire. | 160 |
| paramètre | |
| créer | 55 |
| définition | 155 |
| opération | |
| UML | 55 |
| paramétrée | |
| classe | 82 |
| participation | |
| créer | 26 |
| définition | 155 |
| diagramme de cas d'utilisation | 25 |
| initiateur | 26 |
| multiplicité | 27 |
| partition | |
| UML. | 123 |
| persistance | |
| classe | 42 |
| définition | 155 |
| persistant | |
| attribut | 49 |
| point d'extension | |
| cas d'utilisation. | 29 |
| polymorphe | |
| opération | 53 |
| post-condition | |
| opération | 56 |
| powertype | |
| stéréotype de classe | 44 |
| précision comportementale | |
| état | 115 |
| pré-condition | |
| opération | 56 |

| | |
|---------------------|----|
| protégée | |
| opération | 53 |

Q

| | |
|----------------------|-----|
| qualificatif | |
| définition | 155 |
| rôle | 70 |

R

| | |
|------------------------------------|-----|
| racine | |
| classe | 42 |
| rechercher | |
| classe | 40 |
| référencer | |
| paquetage. | 81 |
| réflexive | |
| association | 72 |
| relier | |
| paquetage | |
| <i>cas d'utilisation</i> | 25 |
| réorganiser | |
| dessin | 94 |
| retailer | |
| objet | 22 |
| rôle | |
| agrégation | 67 |
| association | 62 |
| composition | 67 |
| définition | 155 |
| diagramme d'objet. | 88 |
| instance | 88 |
| lecture seule | 68 |
| lien | 88 |
| modifiable. | 68 |
| multiplicité | 63 |
| <i>diagramme d'objet</i> | 88 |
| navigabilité | 66 |
| ordre | 69 |
| qualificatif. | 70 |
| statique | 69 |

S

| | |
|-----------------------|-----|
| schema group | |
| stéréotype de classe | 44 |
| sélecteur | |
| stéréotype | 52 |
| séquence | |
| diagramme de séquence | 131 |
| séquentielle | |
| opération | 53 |
| signal | 58 |
| définition | 155 |
| diffusion | 59 |
| stéréotype | 59 |
| type expression | 59 |
| visibilité | 59 |
| vote | 59 |
| signature | 53 |
| contrôle | 54 |
| opération | 53 |
| signal | 53 |
| statique | |
| attribut | 49 |
| opération | 52 |
| rôle | 69 |
| stéréotype | |
| classe | 42 |
| définition | 155 |
| option d'affichage | 45 |
| structure | |
| stéréotype de classe | 44 |
| swimlane | |
| UML | 123 |

T

| | |
|----------------------|-----|
| taille | |
| objet | 22 |
| temporisateur | |
| définition | 155 |
| ternaire | |
| association | 72 |
| textes | |
| objet | 43 |
| thread | |
| stéréotype de classe | 44 |
| transition | |
| affichage | 118 |
| définition | 156 |
| externe | 117 |

| | |
|-------------------|-----|
| interne | 117 |
| <i>définition</i> | 156 |
| locale | 118 |
| type | 117 |
| UML | |
| <i>affichage</i> | 118 |

type

| | |
|----------|------------|
| attribut | 47, 49, 50 |
|----------|------------|

type de données

| | |
|------------|-----|
| définition | 156 |
|------------|-----|

type élémentaire

| | |
|-----------|-----|
| composé | 164 |
| définir | 158 |
| nouveau | 163 |
| paquetage | 160 |

type expression

| | |
|------------|-----|
| définition | 156 |
|------------|-----|

U

utilisation

| | |
|------------|-----|
| définition | 156 |
|------------|-----|

utilitaire

| | |
|----------------------|----|
| stéréotype de classe | 44 |
|----------------------|----|

V

valeur

| | |
|----------|----|
| attribut | 87 |
|----------|----|

visibilité

| | |
|-----------|--------|
| attribut | 47, 50 |
| classe | 42 |
| opération | 53 |

Z

zoom

| | |
|-----------|----|
| diagramme | 27 |
|-----------|----|