

UML : Les bases

Table des matières



Objectifs	3
Introduction	4
I - Le langage UML : définitions, objectifs et histoire	5
1. Définition UML	5
2. Les objectifs d'UML	7
3. Histoire d'UML	7
3.1. La période de fragmentation :	8
3.2. La période d'unification	8
3.3. La période de normalisation	8
3.4. La période de révision	8
3.5. La période d'industrialisation	9
4. Quand utilisé UML ?	9
5. Exercice : Exercices	10
II - Le langage UML : Généralité sur les diagrammes	12
1. Les diagrammes UML	13
1.1. Les catégories de diagrammes UML	13
1.2. 3. Diagrammes interactifs	15
1.3. Conclusion	16
2. Exercice : Exercices	16

Objectifs

A la fin de cette leçon vous serez capable de :

- *Définir* le langage UML ;
- *Lister* les principaux diagrammes UML

Introduction



Au cours de la leçon 1, nous nous sommes arrêté sur comment modéliser un logiciel ? En répondant à cette question, nous avons introduit la notion du langage UML .

Mais au fond qu'est ce que UML ? D'où vient ce nom UML ? Pourquoi c'est l'UML qui a été choisi pour modéliser les logiciels ? Comment le langage UML fonctionne ?

C'est à ces questions que nous allons répondre au cours de cette leçon.



Figure 1 : Logo officiel du Langage UML

Le langage UML : définitions, objectifs et histoire

I

Objectifs

A la fin de cette section vous serez capables de :

- Définir le langage UML
- Connaître l'origine du langage UML
- Définir le mode de fonctionnement du langage UML

1. Définition UML

Définition

UML est l'acronyme anglais de « *Unified Modeling language* ». En Français *Langage de modélisation unifié*.

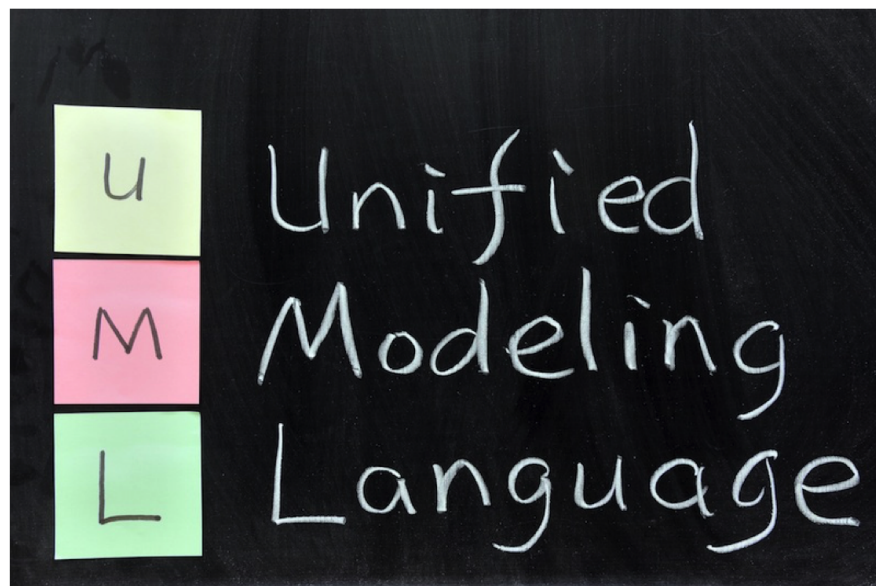


Figure 2 : Définition de l'acronyme UML

Le langage UML est un *langage graphique de modélisation* initialement conçu pour *représenter, spécifier, concevoir et documenter* les *artefacts* de systèmes logiciels. Il est devenu une référence incontournable dans le domaine du génie logiciel.

UML résulte de l'unification de techniques ayant fait leurs preuves pour l'analyse et la conception de grands logiciels et de systèmes complexes.¹

 Complément : Définition des artefacts dans le développement logiciel.

Artefact est un terme général désignant toute sorte d'information créée, produite, modifiée ou utilisée par les travailleurs dans la mise au point du système. Plus d'explications *ici*

 Définition : Définition de UML selon Wikipédia :

Le langage de modélisation unifié, de l'anglais Unified Modeling Language (UML), est un langage de modélisation graphique à base de *pictogrammes* conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. Il est couramment utilisé en *développement logiciel* et en *conception orientée objet*.

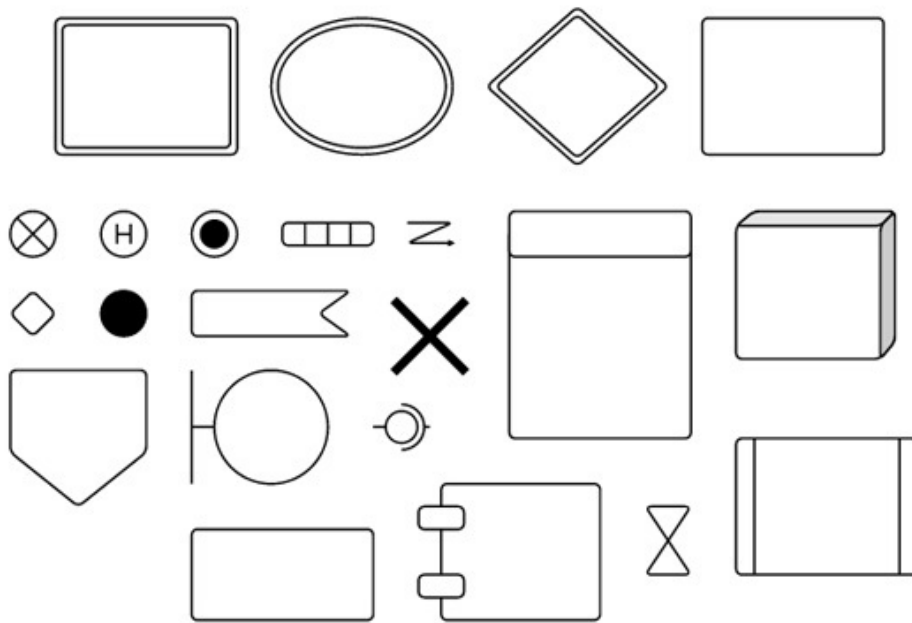


Figure 3 : Exemple de pictogrammes UML

Définition : Autres Définitions de UML

UML est un *langage visuel* constitué d'un ensemble de *schémas*, appelés des *diagrammes*

UML nous fournit donc des *diagrammes* pour représenter le logiciel à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par le logiciel

UML est *unifié* car il résulte de l'unification de techniques ayant fait leurs preuves pour l'analyse et la conception de grands logiciels et de systèmes complexes

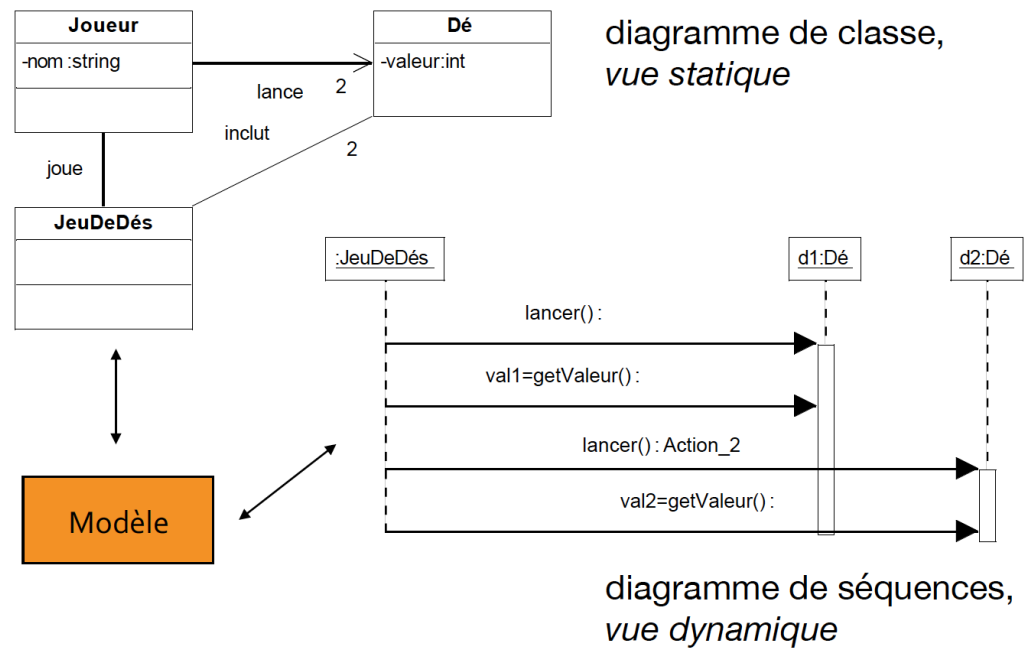


Figure 4 : Exemple de diagrammes UML

2. Les objectifs d'UML

- UML est standardisé par l'OMG (Object Management Group), est un formalisme graphique de modélisation, pour représenter l'architecture logicielle
- UML permet de décrire et concevoir des systèmes logiciels
- UML sert à :
 - Décomposer le processus de développement
 - Mettre en relation les experts métiers et les analystes,
 - Coordonner les équipes d'analyse et de conception,
 - Migrer facilement vers une architecture
 - Rendre indépendant des langages de programmation

Complément : OMG

L'Object Management Group (OMG) est une association américaine à but non lucratif créée en 1989 dont l'objectif est de standardiser et promouvoir les *technologies objet* sous toutes ses formes. <http://www.omg.org>

3. Histoire d'UML

L'histoire d'UML se compose de 5 périodes distinctes :

3.1. La période de fragmentation :

Entre le milieu des années 1970 et le milieu des années 1990, les organisations commencèrent à saisir la valeur du logiciel pour les entreprises, mais ne disposaient que d'une collection fragmentée de techniques permettant de le produire et de l'entretenir. Parmi les diverses techniques émergentes et les méthodes se concentrant sur la production et la maintenance efficace des logiciels (chacune d'elles possédant ses propres langages de modélisation), trois se démarquèrent :

1. *La méthode BOOCH* de Grady Booch mettait l'accent sur le design et la construction des systèmes logiciels.
2. *La méthode OMT* (Objet Modeling Technique) de James Rumbaugh insistait sur l'analyse des systèmes logiciels.
3. *La méthode OOSE* (Object-Oriented Software Engineering) de Ivar Jacobson se consacrait au business engineering et à l'analyse des exigences.



Grady Booch



James Rumbaugh



Ivar Jacobson

Figure 5 : Auteurs des méthodes à l'origine d'UML

Alors que les méthodes orientées objet commençaient à se développer sur la base des méthodes structurées, l'industrie s'est fragmentée entre ces trois et d'autres méthodes.

3.2. La période d'unification

James Rumbaugh puis Ivar Jacobson rejoignirent Grady Booch chez Rational Software Corporation afin d'unifier leurs approches. Comme les organisations commençaient à saisir la valeur d'UML, le groupe de travail de L'OMG publia une requête RFP (Request for proposal) afin d'établir un standard(une norme) définissant la signification des concepts orientés objet. Rational Software Corporation forma le Consortium des Partenaires UML avec diverses autres organisations et soumit à l'OMG la version 1.0 d'UML en réponse à cette requête. *UML 1.0* émergea entre le milieu des années 1990 et 1997.

3.3. La période de normalisation

UML 1.1 émergea durant la deuxième moitié de 1997. Toutes les réponses à la requête RFP furent combinées dans la version 1.1 du langage.

L'OMG adopta UML et assumait en novembre 1997 la responsabilité du développement futur de ce standard.

3.4. La période de révision

Plusieurs versions se succédèrent après l'adoption d'UML 1.1. L'OMG forma un groupe de révision chargé de recevoir les commentaires du public et d'apporter des corrections mineures au standard.

3.5. La période d'industrialisation

En parallèle à cette période de révision, l'OMG soumet *UML* à une normalisation internationale auprès de l'organisme ISO (International Organisation for Standardization).

3.6

.

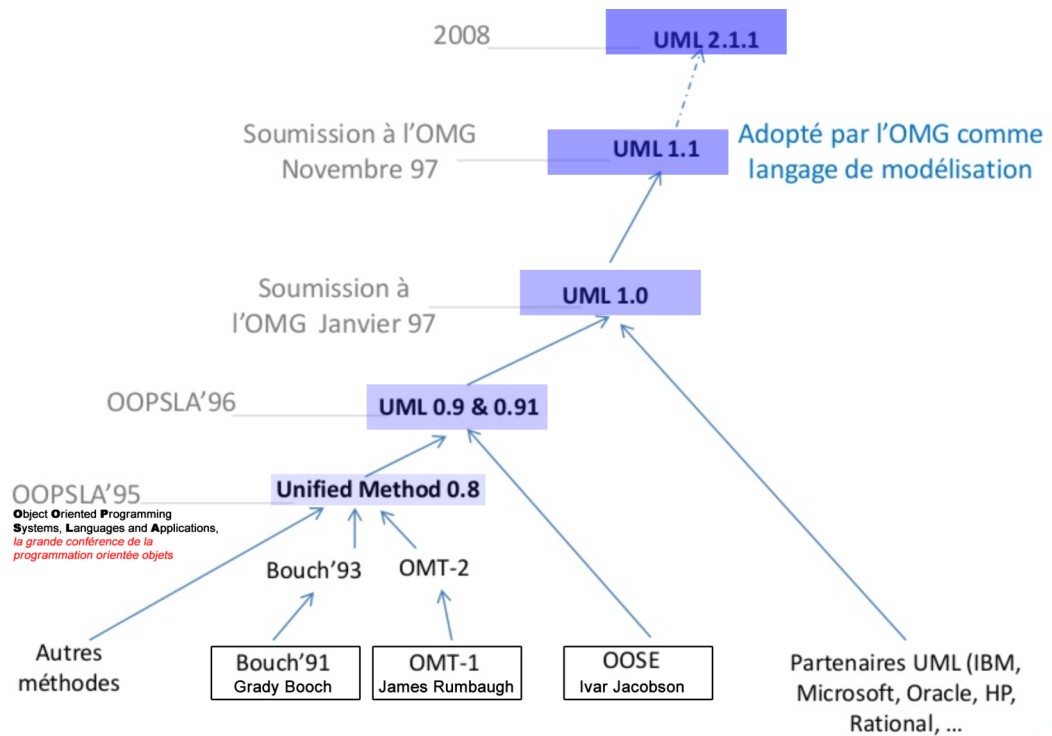


Figure 6 : Organigramme de l'évolution du langage UML

4. Quand utilisé UML ?

UML est surtout utilisé lorsqu'on prévoit de développer des applications avec un langage de programmation orienté objet.

La figure 7 indique la zone de couverture de la modélisation UML dans la programmation informatique

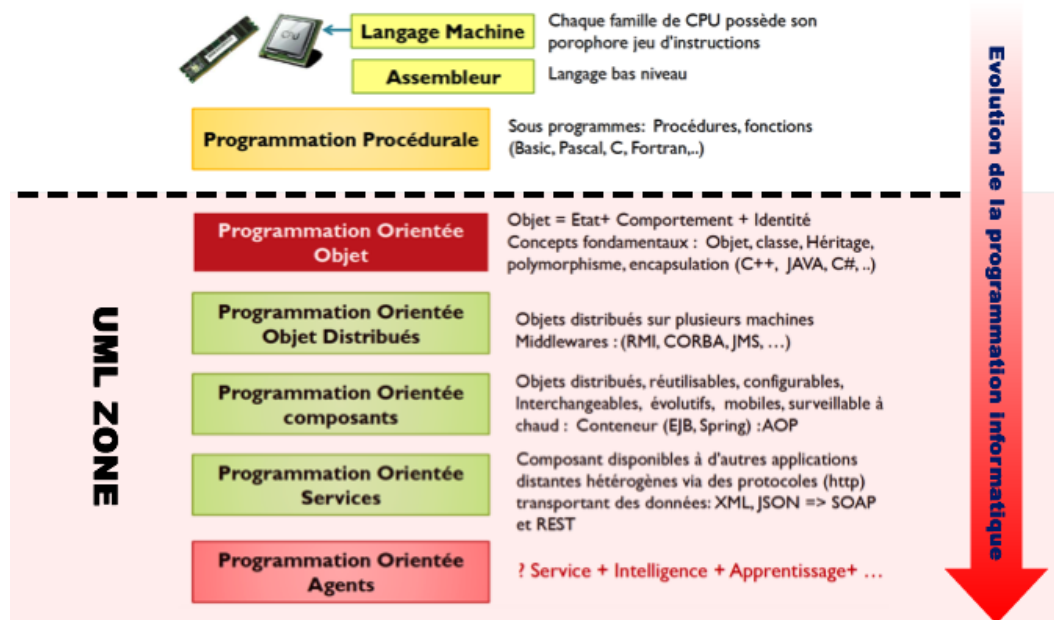


Figure 7 : la programmation informatique : zone couverte par UML

5. Exercice : Exercices

Exercice : Exercice 01

Quelles sont les méthodes de modélisation qui ont fusionnées pour créer UML ?

- ☐ La méthode BOOCH
- ☐ La méthode OUSE
- ☐ La méthode OOSE
- ☐ La méthode OMG
- ☐ La méthode ONT
- ☐ La méthode OMT

Exercice : Exercice 02

Que signifie UML ?

- ☐ UML : Unify Modeling Language
- ☐ UML : United Modele Language
- ☐ UML ; Unified Modeling Language

Exercice : Exercice 03

En quelle année UML est devenu un langage de modélisation officiel?

- ☐ ISO

- ☐ OSI
- ☐ OMG

Le langage UML : Généralité sur les diagrammes



Objectifs

A la fin de cette section vous serez capables de :

- *Lister* les types de diagrammes UML
- *Lister* les principaux diagrammes UML

Le langage de modélisation UML permet de modéliser les exigences des applications à développer à l'aide de différents types de diagrammes. chaque diagramme représente graphiquement le point de vue particulier du logiciel

1. Les diagrammes UML

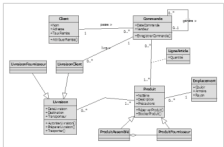
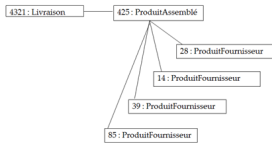
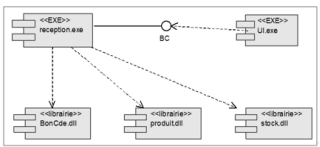
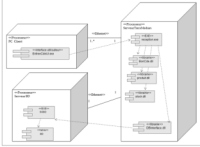
1.1. Les catégories de diagrammes UML

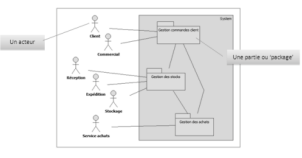
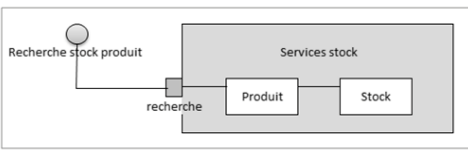
Selon le site web officiel www.uml.org, UML, dans sa version 2.0 définit treize (13) types de diagrammes repartie en trois (3) catégories. :

1. Diagrammes *structurels*
2. Diagrammes *comportementaux*
3. Diagrammes *interactifs*

1.1.1. 1. Diagrammes structurels

Les diagrammes structurels permettent de présenter une vue des éléments statiques du systèmes : les objets en présence, les classes d'objets, les composants... Il n'est pas ici question de modéliser de quelle manière le système se comporte, il est constitué d'un sous-ensemble de diagrammes . :

Nom du diagramme	Description du diagramme	Exemple du diagramme
Diagramme de <i>classe</i>	Il est destiné à décrire les propriétés structurelles des objets du monde réel,	
Diagramme d' <i>objets</i>	Il offre un moyen de représenter les objets (c'est-à-dire les instances des classes figurant dans les diagrammes de classes) ainsi que leurs relations ;	
Diagramme de <i>composants</i>	Il est destinés à la description des éléments de configuration qui constituent le logiciel et à la formalisation de leurs dépendances	
Diagramme de <i>déploiement</i>	Il permet de représenter l'implantation des différents programmes et composants logiciels sur l'architecture physique du système	
Diagramme de <i>paquetages</i>		

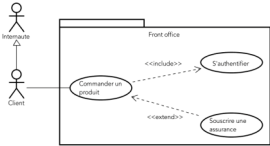
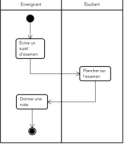
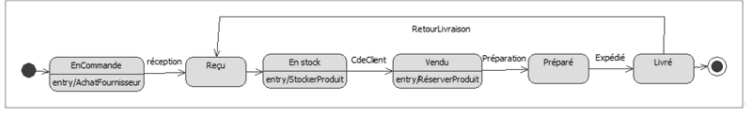
	Il est la représentation graphique des relations existant entre les paquetages composant un système	
Diagramme de <i>structure composite</i>	Il expose la structure interne d'une classe ainsi que les collaborations que cette dernière rend possible	

Définition

- *Une Classe* : Une classe est un « moule » à objets, la description d'un ensemble d'objets par leur caractéristiques
- *Une Instance* : Une instance est un exemple concret de contenu d'une classe
- *Une structure composite* est un ensemble d'éléments interconnectés collaborant dans un but commun lors de l'exécution d'une tâche. Chaque élément se voit attribuer un rôle dans la collaboration

1.1.2. 2. Diagrammes comportementaux

Les diagrammes de comportement permettent de spécifier le comportement des éléments d'un système, et leur dynamique. On les utilise notamment pour modéliser des processus et des événements chronologiques. il est constitué d'un sous-ensemble de diagrammes .

Nom du diagramme	Description du diagramme	Exemple du diagramme
Diagramme de <i>cas d'utilisation</i>	Il est utilisé pour donner une vision globale du comportement fonctionnel d'un système logiciel.	
Diagramme d' <i>activité</i>	Il permet de modéliser un processus interactif, global ou partiel d'un logiciel.	
Diagramme d' <i>états-transitions</i>	Les diagrammes d'états-transitions d'UML décrivent le comportement interne d'un	

	objet à l'aide d'un automate à états finis.	
--	---	--

Complément

Un automate : Un automate à états finis est graphiquement représenté par un graphe comportant des états.

Exemple : une ampoule a 2 états , soit elle est allumée ou éteinte.

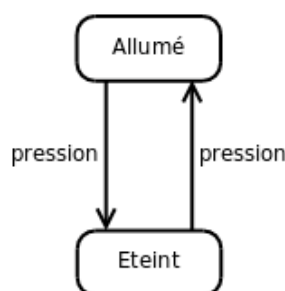


Figure : automate à états finis

Cet automate possède deux états (Allumé et Éteint) et deux transitions correspondant au même événement : la pression sur un bouton d'éclairage domestique.

1.2. 3. Diagrammes interactifs

Les diagrammes interactifs sont un sous-ensemble du grand ensemble de diagrammes comportementaux. il est composé de des sous diagrammes suivants :

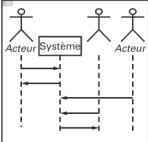
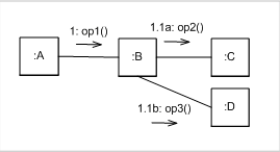
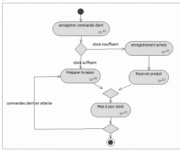
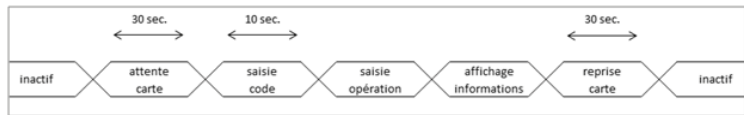
Nom du diagramme	Description du diagramme	Exemple du diagramme
Diagramme de séquence	Il est la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique	
Diagramme de communication	c'est une représentation simplifiée d'un diagramme de séquence se concentrant sur les échanges de messages entre les objets	

Diagramme <i>global d'interaction</i>	Ce diagramme est apparu dans la version 2 de UML, Le diagramme global d'interaction ou diagramme d'interactivité est utilisé pour rendre compte de l'organisation spatiale des participants à l'interaction.	
Diagramme de <i>temps</i>	Un diagramme de temps est un diagramme d'interaction où l'attention est portée sur les contraintes temporelles.	

1.3. Conclusion

Nous allons aborder principalement dans ce cours un sous-ensemble de diagrammes fréquemment utilisés :

Diagrammes structurels

- Diagramme de *classes*
- Diagramme d'*objets*
- Diagramme de *composants*

Diagrammes comportementaux

- Diagramme de *cas d'utilisation*
- Diagramme d'*activité*

Diagrammes interactifs

- Diagramme de *séquence*

2. Exercice : Exercices

Exercice : Exercice 01

Quelles sont catégories UML ?

- ☐ Diagrammes de liaison

- ☐ Diagrammes interactifs
- ☐ Diagrammes de classe
- ☐ Diagrammes comportementaux
- ☐ Diagrammes communicants
- ☐ Diagrammes transactionnels
- ☐ Diagrammes structurels

Exercice : Exercice 02 - 1/2

Cochez les bonnes réponses.

- ☐ Diagramme de séquence
- ☐ Diagramme de communication
- ☐ Diagramme global d'interaction
- ☐ Diagramme de temps
- ☐ Diagramme de transitions
- ☐ Diagramme d'états-transitions
- ☐ Diagramme d'activité
- ☐ Diagramme d'utilisation
- ☐ Diagramme de composite
- ☐ Diagramme de séquence

Exercice : Exercice 02 - 2/2

Cochez les bonnes réponses.

- ☐ Diagramme de classe
- ☐ Diagramme d'objets
- ☐ Diagramme de composites
- ☐ Diagramme de composants
- ☐ Diagramme de déploiement
- ☐ Diagramme de paquets
- ☐ Diagramme de structure composite

- ☐ Diagramme d'études
- ☐ Diagramme de cas d'utilisation