Développez vos talents
Certifiez vos compétences

Conception et langages

UML - Analyse et conception

3 jours (21h00) | ★★★★½ 4,3/5 | UML-CO | Code Certif Info : 95453 | Certification M2i Fondamentaux de la Programmation (non incluse) | Évaluation qualitative de fin de stage | Formation délivrée en présentiel ou distanciel [1]

Formations Informatique › Langages et développement › Conception et langages

À l'issue de ce stage vous serez capable de :

- Concevoir des applications objets avec UML
- Comprendre ce qu'est un design pattern
- Connaître les différents diagrammes
- Analyser un problème et le représenter avec UML
- Formaliser les exigences sous forme de use cases
- Détailler les interactions entre objets avec les diagrammes UML
- Utiliser les dossiers de conception rédigés en UML.

Niveau requis

La connaissance d'un langage de programmation est un plus.

Public concerné

Chefs de projets, développeurs, analystes et concepteurs.

Cette formation :

- Est animée par un consultant-formateur dont les compétences techniques, professionnelles et pédagogiques ont été validées par des diplômes et/ou testées et approuvées par l'éditeur et/ou par M2i Formation
- Bénéficie d'un suivi de son exécution par une feuille de présence émargée par demi-journée par les stagiaires et le formateur.

(1) Modalité et moyens pédagogique :

Formation délivrée en présentiel ou distanciel * (e-learning, classe virtuelle, présentiel à distance). Le formateur alterne entre méthodes ** démonstrative, interrogative et active (via des travaux pratiques et/ou des mises en situation). La validation des acquis peut se faire via des études de cas, des quiz et/ou une certification.

Les moyens pédagogiques mis en oeuvre (variables suivant les formations) sont : ordinateurs Mac ou PC (sauf pour les cours de l'offre Management), connexion internet fibre, tableau blanc ou paperboard, vidéoprojecteur ou écran tactile interactif (pour le distanciel). Environnements de formation installés sur les postes de travail ou en ligne. Supports de cours et exercices.

* Nous consulter pour la faisabilité en distanciel. ** Ratio variable selon le cours suivi.

Programme

Jour 1

Introduction

- Apports d'une méthode de modélisation
- Historique
- La normalisation d'UML
- Apports d'UML 2

Difficultés du développement logiciel

- Etat des lieux : les difficultés
- Approches et solutions

Concepts objets

- Approche procédurale et décomposition fonctionnelle
- La transition vers l'approche objet
- Les objets
- Les classes et leurs relations
- Généralisation et hiérarchies de classes
- Le polymorphisme
- Interfaces
- Patrons et classes génériques
- Exceptions

Exemples de travaux pratiques (à titre indicatif)

- *Sur la base d'entités facilement identifiables en entreprise (client, commande), identifier ce qui est éligible à devenir objet et en déduire les classes*
- *Identifier les propriétés des objets trouvés, les verbes, les opérations ou les méthodes qu'ils présentent*

UML et le développement du logiciel

- La nécessité de structurer le développement applicatif
- Cycles de développement logiciel
- UML et le cycle en V
- UML dans les développements itératifs

Jour 2

Diagrammes UML

- Types de diagrammes et éléments communs
- Notes
- Stéréotypes, contraintes et valeurs marquées
- Paquetages
- Relations

Cas d'utilisation (use cases)

- Qu'est-ce qu'un cas d'utilisation ?
- Acteurs et use cases
- Représenter les use cases
- Organisation des use cases

Exemples de travaux pratiques (à titre indicatif)

- *Mise en place d'un projet sous forme textuelle*
- *Lecture et analyse pour trouver les cas d'utilisation, les acteurs, les systèmes et les sous-systèmes*
- *Optimisation de ces cas d'utilisation*
- *Ecriture de scénarios de cas d'utilisation et analyse de ceux-ci pour mesurer les entités à déduire*

Le modèle objet statique

- Diagrammes de classes
- Diagrammes d'objets
- Diagrammes de composants
- Diagrammes de déploiements
- Diagramme de structures composites (UML 2)

Exemples de travaux pratiques (à titre indicatif)

- *A partir des scénarios et use cases précédent, identification des classes principales et de leurs relations*
- *Recherche des propriétés*
- *Réalisation d'un diagramme d'objet pour les relations un peu complexes*
- *Pour entrevoir l'exploitation de la future application et des relations fortes entre classes, des diagrammes de composants et de déploiements sont créés*

Jour 3

Le modèle dynamique

- Diagrammes d'interactions
- Diagrammes d'activités
- Diagrammes d'états transitions
- Les diagrammes de vue d'ensemble d'interactions (UML 2)
- Les diagrammes de timing (UML 2)

Exemples de travaux pratiques (à titre indicatif)

- *Les différents types de classes ou d'objets sont placés dans un diagramme de séquence, permettant de faire vivre la future application et d'identifier les responsabilités de chaque objet*
- *Réalisation d'un diagramme de communication pour répartir les objets sur les couches*
- *Création, sur la base d'un processus métier simple, d'un diagramme d'activité, permettant d'affecter des activités à des rôles et de comprendre l'ordre des activités lors du déroulement d'un processus*
- *Il est choisi une entité dans le projet en cours, et un diagramme d'état est réalisé, pour comprendre la vie complète de cette entité, ce qui la fait passer d'un état à un autre*
- *Revue complète du projet et critique*

Certification (en option)

- Prévoir l'achat de la certification en supplément
- L'examen (en français) sera passé le dernier jour, à l'issue de la formation et s'effectuera en ligne
- Il s'agit d'un QCM dont la durée moyenne est d'1h30 et dont le score obtenu attestera d'un niveau de compétence

Modalités d'évaluation des acquis

L'évaluation des acquis se fait :

- En cours de formation, par des études de cas ou des travaux pratiques

– Et, en fin de formation, par un questionnaire d'auto-évaluation ou une certification (M2i ou éditeur)

Compétences visées

- Concevoir et créer des solutions algorithmiques pour résoudre un problème donné
- Traduire de l'algorithme dans un langage de programmation
- Identifier les apports de la modélisation (UML)
- Identifier les variables et le typage des données
- Identifier les bases d'un langage de programmation.