Formation Du procédural à l'objet : concepts, UML et Design Patterns

Durée :	4 jours
Public :	Développeurs et analystes programmeurs (ne connaissant pas l'objet), chefs de projets.
Pré-requis :	Notions d'algorithmique et de programmation procédurale dans un langage quelconque
Objectifs :	Comprendre les enjeux de la conception par objets - Maîtriser les concepts généraux et pouvoir les appliquer aux principaux langages objets - Modéliser une application avec - Maîtriser les concepts de la programmation orientée objet - Acquérir les notions fondamentales pour la modélisation d'un projet en UML - Découvrir les bonnes pratiques d'architecture de code et choisir/implémenter des patrons de conception
Sanction :	Attestation de fin de stage mentionnant le résultat des acquis
Taux de retour à l'emploi:	Aucune donnée disponible
Référence:	PRO1045-F
Note de satisfaction des participants:	Pas de données disponibles

Découvrir la programmation orientée objet

Styles de programmation : impératif, procédural, orienté objet
Comparaison des styles de programmation, apports
Impossibilités et enjeux : passer du procédural à l'objet

Apprendre l'objet

Les classes, attributs et méthodes : éléments fondamentaux
Les instances de classe (objets)
Staticité et dynamicité : correspondance avec la vie réelle
Héritage : réutilisation du code et redéfinition de méthodes
Gestion de la visibilité : facteur, contrôle
Interfaces et abstraction : préparation raisonnée d'une architecture

Atelier : modélisation objet de plusieurs scénarios dans le cadre d'une application e-commerce pour illustrer les différents concepts objet : agrégation, héritage, abstraction et polymorphisme.

Introduction à la modélisation UML

Besoin de modélisation : analyse et conception d'un projet informatique
Présentation du langage : principe, historique et utilité
Démarches de modélisation : UML et les méthodes d'analyse (Merise, Unified Process)
Positionnement des diagrammes dans le cycle de développement.

Atelier : Comparaison des démarches et panorama d'outils de modélisation UML - Terminologie UML et représentation graphique sous forme papier / avec un outil.

UML : Recueil et analyse des besoins

Diagramme des cas d'utilisation : présentation, fonctionnalités
Description des éléments du diagramme : acteurs, cas d'utilisation

Atelier : Modélisation UML d'un système de prise de RDV.

UML : Conception globale (architecturale)

Diagramme de séquence : interactions entre objets au cours du temps. Messages synchrones et asynchrones
Diagramme de composants : description des modules de l'application et description des dépendances

Atelier : Modélisation UML d'un système de commande.

UML Conception détaillée

Diagramme de paquetages : organisation des différentes classes/couches de l'application
Diagramme de classes : représentation statique de la structure interne de l'application
Diagramme d'objets : représentation de l'état du système à un instant donné (expression des exceptions)
Diagramme d'activités : modélisation du flux objet/activité pour la réalisation d'une opération
Diagramme d'états-transitions : détail des transitions affectant l'état d'un objet

Atelier : Modélisation UML d'une application métier.

Introduction aux Design Patterns

Présentation : définition, forme
Domaines d'application des patrons de conception
Classification des patterns : création, structure, comportement
Critères de choix et d'application des Design Patterns

Atelier : analyse des définitions de pattern et factorisation par besoin métier.

Patterns de génération d'instances

Factory et Abstract Factory pour la création sous condition
Singleton et dérivé : maîtrise des ressources disponibles

Patterns de structure des données

Le Composite, comment simplifier les listes
La Facade : clarifier un composant

Pattern de comportement

Strategy : l'usine à méthodes
L'iterateur et ses implémentation existantes
Observer : l'événementiel sans événements
Template : introduire des actions spécifiques dans un comportement standard

Ateliers : Analyse du besoin et proposition d'un pattern adéquat ; modélisation UML et implémentation de la solution proposée par le pattern.