Bibiane Gagné-Caron  
Rita Merveille Matsinkou

Programmation Orientée Objets

Groupe : 01  
420-N26

Travail Pratique 2

Travail présenté à :  
Charles Jacob

Département d’informatique  
CEGEP Régional de Lanaudière à Joliette

21/05/2025

# Dictionnaire du diagramme du Générateur

## FormGenerateur

### Description

FormGenerateur est la classe représentant l’interface graphique principale de l’application. Elle permet à l’utilisateur de gérer l’ensemble des composantes d’un scénario de simulation de trafic aérien : création, modification et suppression d’aéroports et d’aéronefs, ajustement des fréquences d’événements, ainsi que le chargement et l’enregistrement de scénarios.  
Elle implémente également l’interface IObservateur, ce qui lui permet de réagir aux modifications du modèle (Scenario), selon le patron de conception *Observer*.

### Héritage

Super-classe : Form

|  |  |  |
| --- | --- | --- |
| Nom | Type | Description |
| m\_controleur | Attribut | Contrôleur responsable de la logique de l’application (ControleurGenerateur). |
| SetControleur(ControleurGenerateur controleur) | Méthode  Public |  |
| btnAeroport\_Click | Méthode Privé | Gère l’ajout d’un nouvel aéroport à partir des champs de saisie. |
| btnCharger\_Click | Méthode Privé | Charge un scénario depuis un fichier sélectionné. |
| btnEnregistrer\_Click | Méthode Privé | Enregistre le scénario courant dans un fichier. |
| modifierAeroport\_Click | Méthode Privé | Permet de modifier un aéroport existant. |
| listAeroport\_SelectedIndexChanged | Méthode Privé | Détecte le changement de sélection d’un aéroport. |
| SupprimerAeroport\_Click\_1 | Méthode Privé | Supprime l’aéroport sélectionné. |
| nouveauScenario\_Click | Méthode Privé | Crée un nouveau scénario vierge. |
| ReinitialiserTextBox | Méthode Privé | Vide les champs de saisie. |
| btnAeronef\_Click | Méthode Privé | Ajoute un aéronef à l’aéroport sélectionné. |
| btnChangerFrequence\_Click | Méthode Privé | Modifie la fréquence d’un événement. |
| typeEvenement\_SelectedIndexChanged | Méthode privée | Réagit à un changement de type d’événement. |
| type\_SelectedIndexChanged | Méthode privée | Réagit à un changement de type d’aéronef. |
| supprimerAeronef\_Click | Méthode privée | Supprime l’aéronef sélectionné. |
| FormGenerateur | Méthode publique | Constructeur de la classe. Initialise l’interface. |
| SetControleur | Méthode publique | Associe un contrôleur à la vue. |
| AfficherScenario | Méthode publique | Affiche les éléments du scénario courant. |
| MettreAJourVue() | Méthode publique | Met à jour l’interface suite à une notification du modèle. |
| AfficherAeroports() | Méthode publique | Affiche la liste des aéroports dans l’interface. |
| AfficherAeronefs() | Méthode publique | Affiche les aéronefs d’un aéroport donné. |

## Interface ISujet

### Description

ISujet est une interface définissant le comportement d’un sujet observable selon le patron de conception **Observer**. Toute classe qui implémente cette interface devient un « sujet » que d'autres objets (observateurs) peuvent surveiller. Elle permet la gestion dynamique d’une liste d’observateurs et assure leur notification lorsqu’un changement d’état pertinent se produit.

Cette interface est essentielle pour découpler les composants de l’application, en particulier entre le modèle (Scenario) et l’interface graphique (FormGenerateur), favorisant une architecture souple et évolutive.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | Attacher(obs : IObservateur) - Permet d’attacher un observateur. |
| Méthode | Detacher(obs : IObservateur) - Permet de détacher un observateur. |
| Méthode | Notifier() - Notifie tous les observateurs inscrits. |

## Classe IObservateur

### Description

IObservateur est une interface représentant un **observateur** dans le patron de conception **Observer**. Toute classe qui implémente cette interface peut être notifiée par un sujet (ISujet) lorsque celui-ci subit un changement d’état.

L’objectif est de permettre aux classes de réagir automatiquement aux modifications d’un modèle (par exemple : mise à jour de l’interface graphique lorsque le scénario change) sans devoir interroger constamment le sujet. Ce mécanisme favorise la séparation des responsabilités et améliore la modularité du code.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | MettreAJour() - Réagit à une mise à jour du sujet observé. |

## Classe ControleurGenerateur

### Description

ControleurGenerateur est responsable de la gestion des scénarios et des éléments associés, y compris les aéroports, les aéronefs et les événements. Il communique avec le gestionnaire de scénario pour effectuer diverses opérations.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | m\_gestionnaire : GestionnaireScenario - Gestionnaire central du scénario. |
| Attribut | formGenerateur : FormGenerateur - Référence à la vue associée. |
| Méthode | ControleurGenerateur(formGenerateur) - Constructeur de la classe. |
| Méthode | EnregistrerObservateur(obs) - Enregistre un observateur pour les mises à jour. |
| Méthode | AjouterAeroport(data) - Ajoute un aéroport au scénario. |
| Méthode | ChargerScenario(cheminFichier) - Charge un scénario depuis un fichier. |
| Méthode | GenererScenario(cheminFichier) - Génère un scénario et l’exporte vers un fichier. |
| Méthode | ModifierAeroport(...) - Modifie les informations d’un aéroport. |
| Méthode | SupprimerAeroport(nom) - Supprime un aéroport existant. |
| Méthode | CommencerNouveauScenario() - Initialise un nouveau scénario vierge. |
| Méthode | AjouterAeronef(...) - Ajoute un aéronef à un aéroport donné. |
| Méthode | SupprimerAeronef(...) - Supprime un aéronef d’un aéroport. |
| Méthode | ChangerFrequence(...) - Change la fréquence d’un événement donné. |
| Méthode | ObtenirFrequence(type) - Récupère la fréquence d’un type d’événement. |
| Méthode | ObtenirFrequences() - Retourne toutes les fréquences du scénario. |
| Méthode | ObtenirListeAeroports() - Retourne tous les noms d’aéroports. |
| Méthode | ObtenirListeAeronefs(nomAeroport) - Liste les aéronefs d’un aéroport. |

## Classe GestionnaireScenario

### Description

GestionnaireScenario est une classe gérée comme un Singleton, garantissant qu'une seule instance est utilisée à tout moment pour gérer le scénario en cours.

### Design Pattern

• Type : Singleton

• Principe : Garantit une seule instance de GestionnaireScenario, accessible via Instance.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | m\_scenario : Scenario - Scénario actuellement chargé. |
| Attribut | \_instance : GestionnaireScenario - Instance unique (Singleton). |
| Attribut | \_lock : object - Objet de synchronisation pour l’accès multi-thread. |
| Méthode | Instance - Retourne l’instance unique du gestionnaire. |
| Méthode | GetScenario() - Retourne le scénario courant. |
| Méthode | SetScenarioActuel(scenario) - Définit un nouveau scénario actif. |
| Méthode | NouveauScenario() - Réinitialise un scénario vide. |
| Méthode | ObtenirListeAeroports() - Retourne les aéroports disponibles. |
| Méthode | ObtenirListeAeronefs(nomAeroport) - Retourne les aéronefs d’un aéroport. |

## Classe Scenario

### Description

Scenario représente un ensemble de données et d'événements permettant de simuler une situation spécifique, incluant des aéroports et des clients.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | m\_observateurs : List<IObservateur> - Liste des observateurs inscrits. |
| Attribut | m\_aeroport : List<Aeroport> - Liste des aéroports du scénario. |
| Attribut | m\_frequence : List<FrequenceEvenement> - Fréquences des événements définis. |
| Méthode | Scenario() - Constructeur de scénario. |
| Méthode | Attacher(obs) - Attache un observateur au scénario. |
| Méthode | Detacher(obs) - Détache un observateur du scénario. |
| Méthode | Notifier() - Notifie tous les observateurs. |
| Méthode | AjouterAeroport(...) - Ajoute un nouvel aéroport au scénario. |
| Méthode | ModifierAeroport(...) - Modifie un aéroport existant. |
| Méthode | SupprimerAeroport(nom) - Supprime un aéroport du scénario. |
| Méthode | AjouterAeronef(...) - Ajoute un aéronef à un aéroport donné. |
| Méthode | SupprimerAeronef(...) - Supprime un aéronef d’un aéroport. |
| Méthode | ObtenirListeAeroports() - Retourne les noms des aéroports. |
| Méthode | ObtenirListeAeronefs(nomAeroport) - Liste les aéronefs d’un aéroport. |
| Méthode | GetFrequence(type) - Retourne la fréquence d’un type d’événement. |
| Méthode | GetFrequences() - Liste les fréquences de tous les événements. |
| Méthode | ChangerFrequence(...) - Modifie la fréquence d’un type d’événement. |

## Classe GestionnaireFichierXML

### Description

GestionnaireFichierXML est une classe utilitaire qui permet d’**importer** et **d’exporter** des scénarios au format XML. Elle centralise la gestion des fichiers de simulation sans avoir besoin d’instanciation multiple.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | Importer(cheminFichier : string) : Scenario — Importe un scénario à partir d’un fichier XML et retourne une instance de Scenario. |
| Méthode | |  | | --- | |  |   Exporter(scenario : Scenario, cheminFichier : string) — Exporte le scénario fourni vers un fichier XML au chemin spécifié. |

## Classe Aeroport

### Description

Aéroport représente un aéroport avec ses caractéristiques principales, y compris sa position, sa capacité de passagers et de cargaisons, ainsi que la liste des aéronefs qui y sont stationnés.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | Nom : string — Nom de l’aéroport. |
| Attribut | Position : Position — Position géographique de l’aéroport. |
| Attribut | MinPassagers : int — Nombre minimal de passagers pouvant être gérés. |
| Attribut | MaxPassagers : int — Nombre maximal de passagers pouvant être gérés. |
| Attribut | MinCargaisons : double — Quantité minimale de cargaisons que l’aéroport peut accueillir. |
| Attribut | MaxCargaisons : double — Quantité maximale de cargaisons que l’aéroport peut accueillir. |
| Attribut | Aeronefs : List<Aeronef> — Liste des aéronefs actuellement stationnés à l’aéroport. |
| Méthode | Aeroport(...) — Constructeur permettant d’instancier un aéroport avec ses propriétés de base. |
| Méthode | AjouterAeronef(...) — Ajoute un aéronef à l’aéroport avec les caractéristiques spécifiées. |
| Méthode | ModifierAeronef(...) — Modifie les propriétés d’un aéronef existant dans l’aéroport. |
| Méthode | SupprimerAeronef(nomAeronef : string) — Supprime un aéronef de l’aéroport selon son nom. |
| Méthode | GetAeronefs() : List<Aeronef> — Retourne la liste complète des objets Aeronef présents dans l’aéroport. |
| Méthode | ToString() : string — Retourne une représentation textuelle de l’aéroport. |
| Méthode | Serialiser() : string — Sérialise les données de l’aéroport en format texte. |
| Méthode | ObtenirListeAeronefs() : List<string> — Retourne les noms des aéronefs de l’aéroport sous forme de texte. |

## Classe FabriqueAeronef

### Description

FabriqueAeronef est une classe singleton qui centralise la création des aéronefs. Elle applique le patron Fabrique pour générer les bons types d’aéronefs selon les paramètres fournis.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | instance : FabriqueAeronef — Instance unique de la fabrique (singleton). |
| Attribut | padlock : object — Objet de verrou pour la synchronisation multithread. |
| Méthode | FabriqueAeronef() — Constructeur privé de la fabrique. |
| Méthode | Instance — Accès à l’instance unique de la fabrique. |
| Méthode | CreerAeronef(...) — Crée un objet Aeronef à partir des paramètres fournis. |

## Classe Aeronef

### Description

Aéronef est une classe abstraite représentant un aéronef générique. Elle définit les attributs de base communs à tous les types d’aéronefs, comme le nom, la vitesse et le temps d’entretien.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | Nom : string — Nom de l’aéronef. |
| Attribut | Vitesse : double — Vitesse de déplacement de l’aéronef. |
| Attribut | TempsEntretien : double — Temps nécessaire à l’entretien de l’aéronef. |
| Attribut | type : TypeAeronef — Type de l’aéronef. |
| Attribut | typeEtat : TypeEtat – donne l’état de l’aeronef |
| Méthode | Aeronef() — Constructeur par défaut. |
| Méthode | Aeronef(...) — Constructeur avec paramètres. |
| Méthode | ToString() : string — Représentation textuelle de l’aéronef. |
| Méthode | Serialiser() : string — Sérialise les données de l’aéronef. |

## Classe AeronefTransport

### Description

AeronefTransport est une classe abstraite dérivée d’Aeronef. Elle représente les aéronefs servant au transport (passagers ou cargo) et ajoute des propriétés liées à l’embarquement, au débarquement et à la capacité.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | Capacite : double — Capacité de transport de l’aéronef. |
| Attribut | TempsEmbarquement : double — Durée d’embarquement. |
| Attribut | TempsDebarquement : double — Durée de débarquement. |

## Classe AeronefUrgence

### Description

AeronefUrgence est une classe abstraite dérivée d’Aeronef, utilisée pour les interventions d’urgence. Elle sert de base pour les aéronefs comme les avions de secours ou les citernes.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | AeronefUrgence() — Constructeur par défaut. |
| Méthode | AeronefUrgence(...) — Constructeur avec paramètres. |
| Méthode | ToString() : string — Représentation textuelle. |
| Méthode | Serialiser() : string — Sérialisation de l’aéronef. |

## Classe AvionPassager

### Description

AvionPassager est un aéronef de type transport conçu pour le déplacement de passagers. Il hérite de AeronefTransport et possède une capacité en nombre de passagers.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | Capacite : int — Nombre de passagers transportés. |
| Méthode | AvionPassager(...) — Constructeur avec paramètres. |
| Méthode | AvionPassager() — Constructeur par défaut. |
| Méthode | ToString() : string — Représentation textuelle. |
| Méthode | Serialiser() : string — Sérialisation. |

## Classe AvionCiterne

### Description

AvionCiterne est un aéronef utilisé pour la lutte contre les incendies. Il transporte des liquides et hérite de AeronefUrgence.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | AvionCiterne() — Constructeur par défaut. |
| Méthode | AvionCiterne(...) — Constructeur avec paramètres. |
| Méthode | ToString() : string — Représentation textuelle. |
| Méthode | Serialiser() : string — Sérialisation. |

## Classe AvionCargaison

### Description

AvionCargaison est un aéronef dédié au transport de marchandises. Il hérite de AeronefTransport et transporte une capacité définie de cargaison.

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | Capacite : double — Capacité de transport de la cargaison. |
| Méthode | AvionCargaison() — Constructeur par défaut. |
| Méthode | AvionCargaison(...) — Constructeur avec paramètres. |
| Méthode | ToString() : string — Représentation textuelle. |
| Méthode | Serialiser() : string — Sérialisation. |

## Classe AvionSecours

### Description

AvionSecours est un aéronef spécialisé dans les missions de sauvetage. Il hérite de AeronefUrgence et intervient rapidement pour des secours.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | AvionSecours() — Constructeur par défaut. |
| Méthode | AvionSecours(...) — Constructeur avec paramètres. |
| Méthode | ToString() : string — Représentation textuelle. |
| Méthode | Serialiser() : string — Sérialisation. |

## Classe Helicoptere

### Description

Helicoptere est un aéronef polyvalent dérivé de AeronefUrgence, utilisé pour les interventions ponctuelles comme l’observation ou le repérage.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | Helicoptere() — Constructeur par défaut. |
| Méthode | Helicoptere(...) — Constructeur avec paramètres. |
| Méthode | ToString() : string — Représentation textuelle. |
| Méthode | Serialiser() : string — Sérialisation. |

## Classe FrequenceEvenement

### Description

FrequenceEvenement représente la fréquence d’apparition d’un type d’événement dans le scénario. Elle contient le type d’événement et la fréquence en occurrences par heure.

|  |  |
| --- | --- |
| Type | Nom et description |
| Méthode | FrequenceEvenement() — Constructeur par défaut. |
| Méthode | FrequenceEvenement(type, frequence) — Initialise la fréquence pour un type donné. |
| Méthode | ToString() : string — Retourne une chaîne représentant l’objet. |
| Méthode | GetTypeAsString() : string — Retourne le type d’événement sous forme de texte. |

## Classe Position

### Description

Position est une classe utilitaire représentant une position géographique avec latitude et longitude. Elle offre aussi une méthode pour convertir les coordonnées en format DMS (degrés, minutes, secondes).

|  |  |
| --- | --- |
| Type | Nom et description |
| Attribut | Latitude : double — Latitude de la position. |
| Attribut | Longitude : double — Longitude de la position. |
| Méthode | Position() — Constructeur par défaut. |
| Méthode | Position(latitude, longitude) — Initialise une position géographique. |
| Méthode | ToString() : string — Retourne une représentation textuelle de la position. |
| Méthode | ConvertirEnDMS(coordonnee, estLatitude) : string — Convertit une coordonnée en format DMS. |

## Classe TypeEvenement

### Description

TypeEvenement est une énumération représentant les différents types d’événements possibles dans un scénario (passager, cargaison, secours, incendie, observation).

# Classe TypeAeronef

### Description

TypeAeronef est une énumération définissant les types d’aéronefs disponibles dans le système : passager, cargo, secours, citerne, hélicoptère.

Classe TypeEtat  
**Type :** Énumération (enum)  
**Description :** Cette énumération définit les différents états possibles d'un objet ou d'une entité dans un contexte lié à l'aviation ou au transport. Elle permet de représenter de manière standardisée les phases importantes d'un cycle opérationnel.

## Tableau des relations

|  |  |  |
| --- | --- | --- |
| Relation | Type UML | Description |
| FormGenerateur ..> ControleurGenerateur | Dépendance | FormGenerateur utilise temporairement ControleurGenerateur. |
| FormGenerateur ..|> IObservateur | Réalisation | FormGenerateur implémente l’interface IObservateur. |
| ControleurGenerateur --> GestionnaireScenario | Association forte | ControleurGenerateur contient une référence à GestionnaireScenario. |
| ControleurGenerateur --> GestionnaireFichierXML | Association | ControleurGenerateur utilise GestionnaireFichierXML pour les opérations de fichier. |
| GestionnaireScenario --> Scenario | Association forte | GestionnaireScenario possède et gère un objet Scenario. |
| Scenario --> Aeroport | Composition | Scenario contient et gère des objets Aeroport. |
| Scenario ..|> ISujet | Réalisation | Scenario implémente l’interface ISujet. |
| Aeroport --> Aeronef | Composition | Aeroport contient et gère des objets Aeronef. |
| FabriqueAeronef --> Aeronef | Dépendance | FabriqueAeronef crée des instances d’Aeronef. |
| AeronefTransport --|> Aeronef | Héritage | AeronefTransport hérite de la classe abstraite Aeronef. |
| AvionPassager --|> AeronefTransport | Héritage | AvionPassager hérite de AeronefTransport. |
| AvionSecours --|> AeronefUrgence | Héritage | AvionSecours hérite de AeronefUrgence. |
| AeronefUrgence --|> Aeronef | Héritage | AeronefUrgence hérite de la classe abstraite Aeronef. |
| AvionCargaison --|> AeronefTransport | Héritage | AvionCargaison hérite de AeronefTransport. |
| AvionCiterne --|> AeronefUrgence | Héritage | AvionCiterne hérite de AeronefUrgence. |
| Helicoptere --|> AeronefUrgence | Héritage | Helicoptere hérite de AeronefUrgence. |
| FrequenceEvenement --\* Scenario | Aggregation | Scenario référence plusieurs objets FrequenceEvenement. |
| Position \*-- Aeroport | Aggregation | Aeroport contient une position géographique (Position). |