

Projet de programmation orientée objet

Livret étudiant – UE XLG3IU030 – L2 informatique – Nantes Université

Rédacteurs : R. Dufour, L. Granvilliers, C. Jermann

Année 2024/2025



Résumé

Ce livret présente le projet de développement logiciel en équipe au sein de l'UE de programmation orientée objet de la L2 informatique de Nantes Université. Il présente les objectifs de ce projet et donne un cadrage précis en termes de calendrier et de tâches à effectuer. Chaque équipe devra s'appuyer sur ce document pour réaliser son projet dont, en particulier, il faudra définir le sujet.

1 Descriptif UE

1.1 Cadre général

La programmation orientée objet (POO) est un paradigme de programmation né dans les années 1970 autour du langage Smalltalk (1972), le premier d'une grande famille incluant, entre autres, C++ (1984), Eiffel (1985), Python (1991), Java (1995) et C# (2002). Un programme est structuré autour d'un **ensemble d'objets qui communiquent entre eux**. Un objet représente une entité du problème traité, par exemple un objet métier (un client, un compte ou un virement dans une application bancaire), un objet informatique (une liste, un périphérique ou un port réseau) ou un objet graphique (une fenêtre, un bouton ou un menu). Développer un logiciel selon une approche objet consiste à identifier les familles d'objets et leurs relations dans la phase d'analyse et de conception, puis à les coder dans un langage de programmation approprié.

Cette unité d'enseignement (UE) de POO s'inscrit dans le cadre de la formation au développement logiciel au sein de la licence informatique. Elle s'appuie d'une part sur l'UE de L1 « introduction au développement logiciel » et d'autre part sur les enseignements d'algorithmique et de programmation de L1 et L2. Elle vise à inculquer les grands principes de la POO (objet, classe, encapsulation, héritage, polymorphisme), à pratiquer la programmation en Java et à expérimenter le développement en équipe. Elle se déroule au premier semestre de la L2 et comporte de manière classique des cours magistraux (8h CM), des travaux dirigés (20h TD) et des travaux pratiques (12h TP). Elle correspond à 4 crédits ECTS (European Credits Transfer System) équivalant à environ 100h de travail par élève (40h en CM/TD/TP et 60h de travail personnel). Elle comporte un **projet informatique** qui constitue la majorité du temps personnel estimé pour chaque élève. Ce livret explique le déroulement et les objectifs du projet.

1.2 Compétences disciplinaires

Les résultats d'apprentissage correspondent à un ensemble de connaissances, de compétences, de comportements que vous approfondirez dans le cadre de l'UE. Ainsi, à la fin de l'UE, vous saurez :

- Identifier une classe, la distinguer d'un objet et décrire le cycle de vie d'un objet ;
- Appliquer le principe d'encapsulation pour protéger les données d'un objet et publier ses traitements externes ;
- Utiliser le principe de composition pour combiner des objets et séparer les traitements ;
- Appliquer le principe d'héritage pour spécialiser les données et traitements d'un objet ;
- Utiliser le mécanisme de polymorphisme pour réaliser des traitements génériques ;
- Concevoir et développer des applications complètes de petite taille dans un environnement de développement intégré ;
- Coder en Java et générer la documentation technique d'un programme.

2 Descriptif du projet

2.1 Objectifs

Le projet de POO est un travail d'équipe qui a pour but de développer un (morceau de) logiciel fonctionnel selon une approche de développement orientée objet. C'est une activité importante dans l'UE à double titre. D'une part, il permet d'approfondir les concepts de la POO et d'expérimenter la programmation en Java en traitant un problème concret sur un temps long. D'autre part, il permet de se confronter au travail collaboratif au sein d'un projet informatique comme dans le monde professionnel. Mauvaise nouvelle pour les *geeks solitaires*, il s'agit de partager du code et des documents, de planifier le projet, de communiquer, de fournir des livrables, etc. Par le passé, le projet était réalisé en binômes dans le cadre d'un sujet donné par l'équipe pédagogique à toute la promotion sur un temps court à la fin du semestre. Mais les anciens élèves ont jeté *Another Brick in the Wall* : pas assez de temps, pas assez de liberté, trop ambitieux ou pas assez !

C'est pourquoi une nouvelle organisation est proposée. La pierre angulaire de cette organisation est que chaque équipe aura la liberté de choisir et de définir son propre sujet parmi un ensemble de thèmes donnés. Mais pour ne pas être amenés à reconstruire un autre mur, il convient de définir un cadre rigoureux pour que les équipes puissent réussir à mener leurs projets à bien en répondant aux objectifs pédagogiques. En effet, comment peut-on définir un sujet de projet intéressant dans une matière en cours d'apprentissage ? Comment doit-on s'organiser au sein d'une équipe informatique ? Quel est le temps imparti au projet et quel est le travail demandé ? Toutes ces questions et bien d'autres encore seront abordées dans les sections suivantes.

2.2 Compétences transversales

En plus des compétences disciplinaires, le projet permettra de développer des compétences transversales autour de la gestion de projet et du travail en équipe. Plus précisément :

- Comprendre les enjeux du travail en équipe ;
- Répartir des tâches au sein d'un projet selon les compétences des individus ;
- Respecter un planning et produire des livrables ;
- Maîtriser un ou plusieurs outils de travail collaboratif ;
- Rédiger un compte-rendu de réunion et un rapport de conception ;
- Présenter les résultats d'un projet sous la forme de diapositives.

2.3 Grille d'évaluation

Le projet sera évalué selon la grille ci-après (sur 2 pages). Une lecture complète du livret est un préalable pour en saisir tous les aspects.

Critère / Évaluation	D : en-deçà des attendus	C : proches des attendus	B : niveau attendu	A : au-delà des attendus
Vue d'ensemble				
Cahier des charges	Sujet peu original, cahier des charges mal rédigé	Sujet assez adapté, cahier des charges partiel	Sujet bien adapté, cahier des charges clair et détaillé	Sujet d'exception, cahier des charges parfait
Interfaces	Interfaces peu soignées, difficile de tester l'application	Interfaces correctes, mais ne donnant pas accès à toutes les fonctionnalités	Interfaces (graphique ou texte) conviviales, facile de tester l'application	Interface graphique complète permettant une utilisation complète du programme
Quantité de travail	Peu de travail fourni, manque de sérieux, intervention en groupe quasi inexistante	Travail mal découpé ou sujet trop ambitieux, organisation à parfaire	Travail découpé en parties avec une juste quantité de classes à développer	Sujet ample adapté à un groupe très performant et bien organisé
Conception et Rapport				
Architecture	Architecture en opposition au cahier des charges	Architecture correcte avec des faiblesses (couplage fort, classes inadéquates)	Diagramme de classes solide, solution extensible	Belle architecture utilisant des patrons de conception (MVC...)
Regard critique	Absence de discours ou mauvaise argumentation	Discours clair sur quelques points, argumentation correcte	Discours clair sur tous les points, bonne argumentation	Un grand recul sur tous les points
Expression écrite	Trame du rapport non respectée, rapport très léger	Trame du rapport respectée, langage à amender, des fautes d'orthographe	Trame du rapport respectée, langage clair, peu de fautes	Rapport excellent à tous niveaux
Programmation				
Code Java	Code obscur, pas de style, algorithmes compliqués	Style en partie respecté, découpage du code et algorithmes perfectibles	Respect du style, méthodes élémentaires, algorithmes corrects	Code professionnel, excellent découpage du code, algorithmes efficaces
Réutilisation	Absence d'utilisation de la librairie standard	Sous-utilisation la librairie standard et/ou redondance de code	Bonne utilisation de la librairie standard (collections, fichiers,...)	Bonne utilisation de la librairie standard et d'autres librairies
Documentation	Pas de documentation	Présence de trous dans la documentation (Javadoc et/ou algorithmes)	Classes et méthodes décrites au format Javadoc et algorithmes commentés	Documentation au niveau de celle du JDK et algorithmes commentés

Critère / Évaluation	D : en-deçà des attendus	C : proches des attendus	B : niveau attendu	A : au-delà des attendus
Gestion de projet				
Organisation équipe	Groupe en conflit, interactions faibles, peu de motivation	Motivation présente mais groupe en sous régime, activité irrégulière	Tâches bien réparties, réunions fréquentes, communication efficace	Groupe très autonome montrant une activité soutenue
Contenu du GitLab	Code source partiel, pas de <i>wiki</i> , mises à jour rares	Mises à jour épisodiques du <i>wiki</i> et/ou absence de code ou de documents	Code source et <i>wiki</i> mis à jour régulièrement, <i>wiki</i> structuré (livrables et CR)	Utilisation extensive du GitLab, rubriques additionnelles dans le <i>wiki</i>
Respect du planning	Nombreux retards et absences	Retards légers pour des livrables ou quelques absences en TD/TP	Fourniture stricte des livrables, présence des groupes en TD/TP	En plus du niveau attendu, CR de toutes les réunions prévues
Soutenance				
Maîtrise du sujet	Le message n'est pas passé	Le message est passé, quelques faiblesses ou imprecisions dans le discours	Exposé clair et adapté montrant une maîtrise des concepts et des techniques	Exposé percutant montrant une grande maîtrise du sujet
Qualité des supports	Trame non respectée, informations partielles, diapositives peu lisibles	Trame respectée, bon niveau d'information, des diapositives à améliorer	Trame respectée, bon niveau d'information, diapositives correctes et lisibles	En plus du niveau attendu, belles diapositives captant l'attention
Expression orale	Exposé difficile montrant un manque de préparation	Ton dynamique, bon niveau de langage mais des hésitations	Ton dynamique, bon niveau de langage, discours relié au support	En plus du niveau attendu, communication très persuasive et argumentée

3 Calendrier et livrables

Le tableau ci-dessous donne le nombre de créneaux de 1h20 dans votre emploi du temps de l'UE sur le semestre. Les colonnes correspondent dans l'ordre aux numéros de semaine, aux cours magistraux, aux travaux dirigés, aux travaux pratiques encadrés et aux travaux pratiques non encadrés. Tous les TP sont consacrés au projet. Les séances de TP non encadrés doivent permettre aux équipes de se connecter à distance aux machines du CIE en mode TP virtuel sous Linux.

Semaine	CM	TD	TP Projet	TP Projet (virtuel)
38	1			
39	1	2		
40	1	2		
41	1	2	1	1
42	1	2	1	1
43	1	2	1	1
44	<i>interruption des cours</i>			
45		2	1	1
46		2	1	1
47		1	1	1
48			1	1
49			1	1
50			1	1
51				

L'organisation du projet dans le temps est présentée dans le diagramme de Gantt ci-après. Il y a cinq phases ou *workpackages* (WP) en anglais, chacune étant conclue par la fourniture d'un livrable (résultat obtenu dans le projet).

1. Le WP1 concerne la lecture de ce document (le livret), la constitution des équipes et la création d'un projet GitLab. Le livrable ❶ est une fiche de compétences individuelle qui devra être remplie par chaque étudiant. La publication des équipes sera faite dans la foulée par les enseignants.
2. Le WP2 consiste à se remuer les méninges pour définir le sujet. Le livrable ❷ est la proposition de sujet de projet remplie.
3. Le WP3 correspond à la phase d'analyse et de conception. Le livrable ❸ est le rapport de conception. Notons que le WP2 et le WP3 se chevauchent car la définition du sujet nécessite de réfléchir à la conception.
4. Le WP4 correspond à la phase de programmation et de test. Le livrable ❹ est le code source.
5. Le WP5 permet de finaliser le rendu. Le livrable ❺ est la présentation sous forme de diapositives. La présentation aura lieu lors d'une dernière séance en présentiel à la fin du semestre.

En plus des cinq livrables identifiés ci-avant, chaque réunion d'équipe doit faire l'objet d'un compte-rendu relatant les sujets abordés et les décisions prises.

Tâche / semaine	38	39	40	41	42	43	44	45	46	47	48	49	50	51
WP1 : lancement														
Lecture du livret	✓	✓												
Fiche de compétences		❶												
Publication équipe			✓											
Création projet GitLab				✓										
WP2 : définition du sujet														
<i>Brainstorming</i>				✓	✓									
Définition gros grain					✓	✓								
Proposition de sujet						❷								
WP3 : conception														
Cahier des charges					✓	✓	✓							
Architecture						✓	✓	✓	✓					
Rapport de conception										❸				
WP4 : programmation														
Codage							✓	✓	✓	✓	✓	✓		
Jeux de tests									✓	✓	✓	✓		
Code source													❹	
WP5 : rendu final														
Préparation des <i>slides</i>												✓	✓	
Soutenance														❺

4 WP1 : lancement

4.1 Constitution des équipes

Vous serez regroupés par équipes de 4 ou 5 personnes, celles-ci étant présentes dans le même groupe de TP (le panachage entre groupes de TP différents est impossible pour des raisons d'emploi du temps). La composition des équipes interviendra de la semaine 39 à la semaine 40. Votre projet sera supervisé par votre chargé de TP qui sera donc votre *chef de projet*. Cet enseignant aura donc accès à tous les documents partagés au sein du groupe, en particulier à travers le GitLab (voir section 4.5).

La constitution de l'équipe est à la charge de votre chef de projet. Il s'appuiera sur la fiche individuelle de compétences que vous aurez à remplir dès le début du module d'enseignement. L'objectif est de regrouper des personnes possédant des compétences complémentaires et d'éviter les déséquilibres.

4.2 Livrable ❶ : Fiche individuelle de compétences

Rendu : remplir le questionnaire en ligne sur Madoc au plus tard le **vendredi 27 septembre 2024 à 18h**. Celui-ci reprend les compétences listées ci-dessous. Les consignes pour le remplir sont à la fois décrites dans ce document ainsi que sur le questionnaire en ligne. Prenez bien le temps de les lire.

La fiche de compétences individuelle vous permet de faire un rapide bilan sur vos forces et faiblesses avant de commencer cette UE. Elle permettra à votre chef de projet de mieux situer vos

compétences et vous même de répartir les tâches à bon escient au sein de votre groupe.

Il est important de la remplir le plus sérieusement possible. Dans le tableau ci-après, vous devez entourer la réponse vous correspondant le mieux à chaque ligne. Veuillez indiquer dans les compétences ci-dessous le niveau qui vous apparaît comme le plus adapté parmi :

- **débutant.** Connaissance limitée des concepts rendant difficile leur application.
- **moyen.** Connaissance avancée des concepts mais aurait encore besoin d'aide pour les appliquer.
- **avancé.** Je maîtrise les concepts et suis capable de les appliquer en autonomie.
- **N/A.** Mes connaissances et/ou mes capacités dans cette compétence sont inexistantes, ou quasi-inexistantes. Ce niveau peut aussi être entouré si la rubrique ne vous concerne pas, ou vous ne savez pas vous situer. Par exemple, vous n'avez pas participé à l'UE xxx.

Attention. Il n'y a pas de bonnes ou mauvaises réponses, cela permet simplement de vous situer au début de l'UE. Soyez donc le plus honnête possible.

Compétences

Bloc 1 - Concevoir un algorithme et le programmer
Moyenne à l'UE Algorithmique et Programmation (X12I010)
Appliquer et analyser des procédures et fonctions
Concevoir des algorithmes de base (calculs, recherches, tris...)
Concevoir des structures de données (tableaux, enregistrements, pointeurs...)
Vérifier l'efficacité des algorithmes
Concevoir des jeux de tests
Bloc 2 - Gérer un projet logiciel
Moyenne à l'UE Introduction au développement logiciel (X12I040)
Gérer un projet avec un outil de versioning de code (ex : GitHub)
Délimiter le projet et l'ampleur du travail
Utiliser des outils de discussion en ligne dans un cadre de travail (Mattermost, Slack, Discord...)
Connaître les principaux enjeux du développement logiciel
Identifier des difficultés et bénéfices de la mise en place d'un projet logiciel
Identifier des critères de qualité d'un code source
Mettre en œuvre une petite application (de la conception à la programmation)

Mener une recherche documentaire complète et de manière efficace
Bloc 3 - Gérer une équipe
Exposer à l'oral de façon claire la problématique et les questions clés d'un projet afin qu'ils soient compréhensibles pour l'équipe
Décrire les solutions possibles à un problème posé
Créer une dynamique d'équipe et motiver le groupe
Planifier un projet (découpage du projet, répartition des tâches...)
Réaliser un compte-rendu détaillé des réunions
Résoudre les conflits
Bloc 4 - Modéliser
Moyenne à l'UE Base de données 1 (X12I030)
Appliquer des concepts d'entité et d'association
Concevoir un modèle entité-association (EA) à partir d'un énoncé
Transcrire un modèle EA vers un schéma relationnel de BDD
Identifier une architecture trois-tiers basée sur un serveur Web, une application et une base de données

4.3 Interactions internes et externes

Dès le début du projet, il sera nécessaire de mettre en place des procédures concernant l'organisation au niveau des interactions au sein de l'équipe (internes), mais également à destination du chef de projet (externes). La manière de s'organiser est libre et doit se faire, dès le départ, en concertation avec votre chef de projet. Cela doit conduire à la mise en place d'un **document écrit**, disponible pour tous et toutes dès la fin de la semaine 40, qui résume cette organisation. À minima, il doit préciser : la fréquence des réunions (au moins hebdomadaire), leurs modalités (en présence et/ou à distance - préciser les outils de communication), les points abordés pendant la réunion ainsi que les personnes/groupes de travail conviés (groupe complet ou restreint).

4.4 Compte-rendu de réunion

Chaque réunion de l'équipe doit être calibrée : date, heure, lieu, objectifs, ordre du jour. Elle fait l'objet d'un compte-rendu (CR) écrit durant la réunion et considéré comme un livrable du projet déposé sur le wiki du projet GitLab. Le CR est un document de travail permettant de faire un état de la discussion et des décisions prises ; les points abordés en réunion peuvent être organisationnels (e.g., qui fait quoi et quand) ou techniques (e.g., un bug observé, un choix d'interface pour un

objet) ; les décisions prises peuvent être des choix (brièvement justifiés), des actions à entreprendre, des points à aborder lors d'une prochaine réunion,...

Le chef de projet pourra exploiter les CR pour effectuer un suivi du projet. Les CR vous seront aussi très utiles pour votre propre suivi du projet et pour la rédaction des livrables. Un exemple de CR vierge (rédigé au format *Markdown*, utilisé par le wiki de Gitlab) est fourni ci-dessous :

```
# Compte-rendu de la réunion du JJ.MM.AAAA
```

```
## Points abordés
```

```
* point A _(quelques détails si besoin)_
```

```
* point B ...
```

```
## Décisions prises
```

```
* décision 1 _(brève justification)_
```

```
* décision 2 ...
```

4.5 GitLab

GIT¹ est un système de gestion de version particulièrement adapté au développement logiciel. Nantes Université déploie un serveur GitLab offrant ce service à l'adresse <https://gitlab.univ-nantes.fr/>. Vous avez appris à utiliser ce service en première année de licence dans le cadre de l'UE *Introduction au développement logiciel* (X12I040, enseignant : G.Sunyé). Si tel n'est pas le cas, ou si vous avez besoin de vous rafraichir la mémoire, vous pourrez trouver des ressources à son sujet via l'aide en ligne du service lui-même².

Vous devez utiliser ce service dans le cadre de votre projet de POO :

- créer, dès la constitution de votre équipe (semaine 41), un projet GitLab qui hébergera votre code source, mais aussi tous les documents/livrables associés ; tous les membres du projet, y compris votre chef de projet, doivent y être inscrits³ en tant que "Maintainer" ;
- mettre à jour, régulièrement et rigoureusement, ces éléments à chaque étape du projet par des *commit* correctement documentés ; organiser les livrables dans des dossiers séparés, avec leurs sources et la version PDF finale bien identifiée ;
- utiliser le wiki associé au projet pour vos compte-rendus de réunions hebdomadaires ; il peut aussi être utilisé pour vos échanges techniques et organisationnels, mais veillez à bien organiser son contenu : la page d'accueil du wiki doit contenir une rubrique *Compte-rendus* listant les liens vers les pages de chaque compte-rendu hebdomadaire ; cette rubrique peut être suivie de rubriques additionnelles (e.g., *Bugs*, *Tâches*, ...) listant à nouveau seulement des liens vers les pages détaillées. Des liens vers le code source du projet et les livrables peuvent être insérés dans les pages wiki lorsque cela est utile.

1. Voir <https://fr.wikipedia.org/wiki/Git>

2. Voir <https://gitlab.univ-nantes.fr/help>

3. Voir le menu "Project Information", sous-menu "Members"

5 WP2 : définition du sujet

5.1 Principes

1. Le sujet doit motiver l'équipe. Ainsi, il faut trouver un domaine d'intérêt en commun : tournoi sportif, réseau logistique, gestion d'une entreprise, jeu, science-fiction, financement participatif, construction de ville, etc.
2. Le sujet doit être original, le plagiat est proscrit.
3. Il faut déterminer le périmètre du sujet, notamment identifier une ou des fonctionnalités à assurer : simuler un tournoi sportif, jouer une partie, simuler une distribution de colis, *etc.* Cela revient à la mise en place d'un cahier des charges.
4. Le sujet doit permettre la mise en œuvre des concepts de la POO. Plus précisément, on veut au moins :
 - plusieurs classes d'objets à développer ;
 - des relations de composition et d'héritage ;
 - du polymorphisme et de la liaison dynamique via des classes abstraites ou interfaces.On peut ajouter à cela le développement d'une interface graphique (voir la section 7.4) sans que ce soit obligatoire (cela dépend du sujet et des envies du groupe).
5. La quantité de travail estimée doit être adaptée à la taille et aux compétences de l'équipe. La calibration du travail sera faite avec l'aide des enseignants en séances de TD et TP.

5.2 Livrable ② : Proposition de sujet

Rendu : déposer un fichier « proposition-sujet.pdf » (au format PDF) dans votre espace GitLab au plus tard le **vendredi 25 octobre à 18h**.

La proposition du sujet est le premier livrable rédigé collectivement par le groupe. Il doit être construit avec la participation de tous les membres et refléter la vision du groupe sur le projet à mener. C'est en quelque sorte le contrat d'engagement de l'équipe autour du sujet. Le sujet doit y être défendu, de façon brève et précise, vis-à-vis des principes exposés ci-avant.

La trame imposée pour la proposition est donnée ci-dessous. Les éléments en *italique* sont des indications que vous pourrez retirer de votre livrable final. Cette trame est également fournie sur madoc aux formats L^AT_EX et PDF.

1. Titre (et acronyme)
2. Lien vers le projet GitLab
3. Composition de l'équipe
Indiquer le nom, prénom, numéro de groupe TP, et adresse mél de chaque membre
4. Compétences de l'équipe
Maximum 1/2 page.
Indiquer les compétences réunies par l'équipe pour réaliser le projet. Pour cela, vous croiserez les fiches individuelles des membres de l'équipe remplies au premier livrable.
5. Résumé du projet
Maximum 10 lignes.
Résumer le thème du projet et les principales fonctionnalités à développer.

6. Positionnement du sujet par rapport aux principes

Maximum 1 page.

Indiquer en quoi le sujet proposé respecte les principes présentés ci-avant.

Vous expliquerez en particulier : votre motivation pour le sujet proposé, son originalité selon vous, les fonctionnalités principales et éventuellement quelques fonctionnalités secondaires ou extensions envisagées, les premiers éléments de conception que vous imaginez (principaux objets/comportements, relations entre ces entités, éléments principaux d'interface utilisateur (graphique ou non), et enfin la répartition envisagée du travail au regard des compétences des membres de l'équipe.

5.3 Thèmes de sujets de projet

Le choix du sujet est libre. Toutefois, une liste de thèmes non exhaustive est proposée ci-après.

- Jeu de type *rogue-like* : un personnage avec différentes compétences et caractéristiques affronte des monstres divers dans des zones de jeu et son niveau de vie évolue en fonction, peut-être jusqu'à la mort.
- Service de livraison : des livreurs avec différentes compétences transportent des colis avec différentes caractéristiques avec une contrainte de temps et un objectif de minimiser des critères comme la trace carbone globale.
- Science fiction : des humains avec différentes caractéristiques se déplacent à bord d'un vaisseau sur différents corps célestes pourvus de ressources dans le but de s'approvisionner ou de fonder une colonie si les conditions le permettent.
- Tournoi sportif : des sportifs possédant chacun un classement et un niveau de jeu s'affrontent en matchs (un contre un) répartis en phases successives (poules, tableau de finales, etc.) et des médailles sont décernées à la fin du tournoi.

6 WP3 : conception

6.1 Objectifs

Le but est de définir l'architecture du programme en fonction du cahier des charges. Le cahier des charges a été décrit brièvement dans la trame de la proposition. Comme il a probablement évolué depuis, il est nécessaire de le mettre au propre. L'architecture du programme consiste en un ensemble de classes (ou interfaces) reliées entre elles et munies de méthodes. Une architecture de qualité doit respecter les grands principes de la POO, en particulier l'encapsulation, l'héritage et le typage dynamique.

Ce travail donne lieu à un rapport dans lequel il faut élucider les principaux aspects de l'architecture, un diagramme de classes permettant d'en avoir une représentation graphique. On demande également un regard critique selon plusieurs critères comme la mise en œuvre des concepts de la POO, le calibrage du projet et la quantité de travail fournie, le travail, la motivation et l'organisation de l'équipe, les compétences de l'équipe au début du projet et le développement de nouvelles compétences, les perspectives d'extension du projet et sur la capacité de la solution actuelle à être extensible.

6.2 Livrable ③ : Rapport de conception

Rendu : déposer un fichier « rapport-conception.pdf » (au format PDF) dans votre espace GitLab au plus tard le **vendredi 22 novembre à 18h**.

La trame imposée pour le rapport est donnée ci-dessous. Les éléments en *italique* sont des indications que vous pourrez retirer de votre livrable final. Cette trame est également fournie sur madoc au formats L^AT_EX et PDF.

Page de garde Elle devra contenir les informations suivantes et respecter cette mise en page :

**Rapport de conception du projet de
programmation orientée objet**

Licence d'informatique – 2^{ème} année

Faculté des sciences et techniques de Nantes

Titre du projet

présenté par

Prénoms et Noms des membres de l'équipe par ordre alphabétique

le *jj mm aaaa*

encadré par

Prénom et NOM du chef de projet



Contenu

1. Cahier des charges
Écrire le cahier des charges du projet : contexte, objectifs, fonctions applicatives, cas d'utilisation. Possible de reprendre des bouts de la proposition de sujet (livrable 2).
2. Architecture
 - (a) Description générale
Décrire avec des phrases les différentes classes et les relations entre ces classes
 - (b) Diagramme de classes
Diagramme de classes UML avec les méthodes principales
 - (c) Interfaces
Description de l'interface console ou de l'interface graphique
 - (d) Aspects spécifiques
Description d'aspects spécifiques au sujet du projet s'il y en a, par exemple des algorithmes non triviaux
3. Regard critique
Sur la mise en œuvre des concepts de la POO
Sur le calibrage du projet et la quantité de travail fournie

Sur le travail, la motivation et l'organisation de l'équipe
Sur les compétences de l'équipe au début du projet et le développement de nouvelles
Sur des perspectives d'extension du projet et sur l'extensibilité de la solution actuelle

6.3 L^AT_EX

L^AT_EX est un système de composition de texte très utilisé dans la communauté scientifique. De nombreux tutoriels sont disponibles⁴. Pour travailler sur un document L^AT_EX à plusieurs, de manière collaborative, il est possible d'utiliser Overleaf à l'adresse <https://www.overleaf.com>. Le document que vous lisez a été produit ainsi. Certains livrables comme la proposition de sujet et le rapport de conception peuvent être écrits en L^AT_EX sans que cela soit obligatoire. À titre d'exemple, la trame du rapport de conception au format L^AT_EX est disponible sur madoc.

7 WP4 : programmation

7.1 Style

La programmation est réalisée en Java. Il convient de respecter un jeu de conventions pour avoir un code cohérent et on propose d'utiliser le *Google style guide*⁵. Le code doit être commenté au format Javadoc (voir ci-après).

7.2 Livrable ④ : Code Java

Rendu : le code est déposé dans votre espace GitLab et il doit être complet au plus tard le **vendredi 13 décembre à 18h** (plus de modifications après cette date).

7.3 Javadoc

Le code doit être commenté au format Javadoc qui est un outil de génération de documentation à partir de commentaires spécifiques écrits dans le code source Java. Ces commentaires adoptent un format particulier et permettent aux développeurs de réutiliser le code développé sous la forme d'un service. Ces commentaires permettent donc de décrire l'utilisation des classes et méthodes développées, mais ne se substituent pas aux commentaires classiques, qui permettent eux de décrire et/ou fournir des informations quant au code écrit (description d'un algorithme, utilité d'une variable...). La Javadoc est à destination des utilisateurs du code développé, alors que les commentaires sont à destination des développeurs du code lui-même.

Un bon exemple de documentation générée par l'outil Javadoc est celle de l'API du JDK décrivant les classes standards de Java. Il existe de nombreux articles et tutoriels sur le sujet⁶.

4. Voir par exemple <https://www.tuteurs.ens.fr/logiciels/latex/>

5. Voir <https://google.github.io/styleguide/javaguide.html>

6. Voir <https://www.oracle.com/fr/technical-resources/articles/java/javadoc-tool.html>

7.4 JavaFX

Attention, il n'est pas demandé de créer une interface graphique, ce qui représente un gros travail à part entière. De plus, il n'est pas prévu de faire un cours sur ce sujet dans l'UE. Une bonne interface texte dans la console est en général suffisante pour ce projet de POO.

Toutefois, si la volonté du groupe est de réaliser une interface graphique, il est conseillé d'utiliser JavaFX⁷, un framework libre (GPL) et multiplateforme permettant la création d'interfaces graphiques en Java, en remplacement des bibliothèques AWT et Swing. Il s'agira donc, dans un premier temps, de regarder la documentation officielle pour bien débiter la programmation JavaFX⁸. La documentation complète est également disponible ici : <https://docs.oracle.com/javase/8/javase-clienttechnologies.htm> et l'API ici : <https://openjfx.io/javadoc/18/>. JavaFX s'intègre bien avec les environnements de programmation intégrés (IDE) tels qu'Eclipse et Netbeans⁹.

Afin de vous familiariser avec le développement d'interfaces avec JavaFX, vous pouvez débiter par les courts tutoriels disponibles ici : https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm. De nombreux autres tutoriels et ressources sont disponibles librement, n'hésitez pas à consulter d'autres ressources. Le site *javatpoint* regroupe par exemples de nombreuses ressources très utiles : <https://www.javatpoint.com/javafx-tutorial>.

7.5 TP virtuels au CIE

Les équipes peuvent se connecter à distance aux machines du CIE sous Linux, en particulier pendant les séances dites de « TP virtuel » présentes dans l'emploi du temps. Pour cela, il faut se connecter à l'URL <https://tpinfo.sciences.univ-nantes.fr/>. La notice d'utilisation du service est disponible sur madoc.

8 WP5 : rendu final

8.1 Descriptif

La soutenance conclut le projet lors de la dernière séance à la fin du semestre. Chaque équipe désigne deux de ses membres qui présentent oralement le projet devant le groupe de TP pendant 10 minutes, avant une phase de questions. Tous les membres de l'équipe doivent impérativement être présents. La soutenance s'appuie sur une série de diapositives montrant le cahier des charges, les réalisations techniques, la gestion du projet et le bilan. En particulier, dans ce bilan, on demande une auto-évaluation par rapport à la grille d'évaluation proposée.

Une diapositive peut comporter des images et du texte devant être lisibles depuis le fond de la salle. Le contenu d'une diapositive guide la personne qui présente et permet au public de suivre la présentation tout en écoutant le discours oral. Une condition est de bien relier le discours au contenu. Une autre condition est d'adapter le contenu au public (ici, le groupe de TP). Par exemple,

7. Voir <https://openjfx.io>

8. Voir <https://openjfx.io/openjfx-docs/>

9. Voir <https://openjfx.io/openjfx-docs/#install-javafx>

il est inutile de rappeler ce qu'est la POO. Un défaut classique est de vouloir tout dire. Au contraire il faut se concentrer sur les aspects essentiels.


Répétez la soutenance et adaptez-la si besoin. Demandez-vous si vous arrivez à faire passer le message (par exemple en invitant d'autres personnes). Attention à respecter le temps imparti !

8.2 Livrable ⑤ : Diapositives

Rendu : un fichier « soutenance.pdf » (au format PDF) doit être déposé dans votre espace GitLab **au plus tard 24h avant la soutenance prévue.**

La trame imposée pour la soutenance est donnée ci-dessous. Les éléments en *italique* sont des indications que vous pourrez retirer de votre livrable final. Pensez à inscrire les numéros de page car cela est utile dans la phase de questions pour revenir à une diapositive précise.

Première diapositive Elle devra comporter les informations suivantes et respecter cette mise en page :

<p>Projet de POO de L2 informatique</p> <p>Titre :</p> <p>présenté par</p> <p>encadré par</p> <p>le <i>jj mm aaaa</i></p> <p> Nantes Université</p>
--

Contenu des autres diapositives

- Contexte
1 à 3 diapositives pour présenter le cahier des charges
- Réalisation technique
2 à 3 diapositives pour présenter l'architecture, les interfaces (en mode console ou graphique), les interactions entre objets, ...
- Gestion de projet
1 à 2 diapositives pour présenter l'organisation de l'équipe (qui a fait quoi, quand, selon quel mode collaboratif / avec quels outils), les livrables fournis
- Bilan
1 à 2 diapositives pour faire un bilan technique (objectifs remplis ? nouvelles technologies apprises ?), un bilan humain (nouvelles compétences acquises ? succès obtenus ? difficultés rencontrées ?), une auto-évaluation du projet par rapport à la grille d'évaluation fournie