Plan de développement

Projet : Conception d'un outil pour améliorer la productivité en préservant la limite du travail humain

BEHAR Andrea
MARTINIANI Anesie
M1 MIASHS- IC

Les informations d'identification du document : ***Les éléments de vérification du document :***

Référence du document :

Version du document : 1.01

Date du document : 12/01/2026

Auteur(s) : Andrea Behar
Anesie Martiniani

Les éléments d'authentification :

Validé par :

Validé le :

Soumis le : 12/01/2026

Type de diffusion : Document électronique (.pdf)

Confidentialité :

Maître d'ouvrage: Chef de projet :

Date / Signature : Date / Signature :

Sommaire

1. Introduction

L'objectif principal de ce projet est de concevoir et de développer un outil permettant la génération automatisée des plans logistiques pour l'ergonomie, tout en répondant à des contraintes multiples (fortes, moyennes ou faibles) et en offrant une flexibilité pour tester différents scénarios. Ce projet est réalisé dans le cadre d'un travail de recherche encadré (TER) par des étudiants de Master MIASHS à l'Université Grenoble Alpes.

L'outil devra répondre aux besoins exprimés par le commanditaire, Mme Landry, et par un utilisateur clé, DPC.

1.1 Objectifs et méthodes

Le développement du logiciel vise à concevoir une solution performante et adaptée pour la génération automatisée d'un plan de logistique ergonomique chez DPC, tout en répondant aux contraintes complexes exprimées par les parties prenantes. Ce projet repose sur une méthodologie Agile, permettant des itérations rapides et des ajustements constants en fonction des retours des utilisateurs et des encadrants.

Les principales étapes de réalisation comprennent :

- Analyse des besoins : Identification des contraintes et attentes des utilisateurs finaux.
- Conception : Développement de l'architecture logicielle et des interfaces utilisateur.
- Développement : Intégration des fonctionnalités principales, notamment l'implémentation du solveur.
- Tests et validation : Validation technique et fonctionnelle à travers des scénarios de tests rigoureux.
- Livraison : Fourniture d'un logiciel complet, accompagné de sa documentation utilisateur et technique.

Cette organisation méthodique assure que chaque phase du projet contribue à la réalisation d'un produit final fiable, performant et conforme aux attentes.

1.2 Documents de référence

Le présent document s'appuie sur un ensemble de ressources clés, essentielles à la conception et au développement du projet. Ces documents ont servi de base pour identifier les besoins, définir les contraintes, et structurer le déroulement du projet. Les documents de référence sont les suivants :

- Cahier des charges (Version 2.02 12.01.2026) : Document de référence décrivant les besoins fonctionnels, les contraintes techniques, et les objectifs du projet.
- Cahier de recette (Version 1.02 12.01.2026) : Document listant les scénarios de tests, les critères de validation, et les modalités de vérification de la qualité du logiciel.

Ces documents garantissent une traçabilité et une cohérence dans toutes les étapes du projet, de sa conception à sa livraison.

2. Guide de lecture

Cette partie à pour objectif de guider les différentes parties prenantes dans l'utilisation optimale des informations fournies dans ce document. Elle détaille les objectifs et les méthodes d'utilisation pour les deux principaux groupes concernés : la maîtrise d'œuvre et la maîtrise d'ouvrage.

2.1 Maîtrise d'œuvre

Les équipes techniques responsables de la conception et du développement de l'outil peuvent utiliser ce document pour :

- Suivre les étapes du projet et garantir la conformité technique des livrables aux exigences définies.
- Identifier et résoudre les anomalies ou problèmes rencontrés lors des tests.
- S'appuyer sur les scénarios de tests et les critères de validation pour vérifier la qualité du produit à chaque phase.

2.2 Maîtrise d'ouvrage

Les décideurs et parties prenantes, en charge de valider les aspects fonctionnels et stratégiques du projet, utiliseront ce document pour :

- Vérifier que les livrables répondent aux besoins exprimés dans le cahier des charges.
- Participer aux phases de validation, notamment via les tests utilisateurs.
- Suivre les résultats des tests et des indicateurs de qualité pour évaluer la pertinence des solutions proposées.

3. Concepts de base

Pour comprendre ce document et les choix techniques réalisés dans le cadre de ce projet, il est essentiel de maîtriser certains concepts clés relatifs à la génération de l'outil et à la gestion des contraintes. Cette section présente les notions fondamentales.

3.1 Contraintes fortes et faibles

Les *contraintes fortes* sont les conditions incontournables qui *doivent impérativement être respectées* pour qu'un plan soit valide. Elles incluent, par exemple, les requis ergonomiques.

Les *contraintes faibles* sont des préférences ou des optimisations souhaitées qui peuvent être partiellement satisfaites en fonction des priorités définies. Par exemple, réduire les temps de trajets pour faire du picking.

3.2 Solveur

Le *solveur* est un *algorithme* spécialisé dans la résolution de problèmes complexes sous contraintes. Il est capable d'optimiser les solutions en fonction des priorités définies. Le projet utilise *LPG*, un solveur *open source*, réputé pour sa robustesse et son efficacité dans la gestion de contraintes complexes.

3.3 Base de données

L'outil s'appuie sur une b*ase de données* conçue pour stocker toutes les informations nécessaires à la génération des emplois du temps, notamment :

- Les listes de professeurs, de classes, et de salles.
- Les contraintes spécifiques associées à chaque ressource.
- Les configurations des scénarios d'emplois du temps.

Cette architecture permet une gestion dynamique des données et offre la flexibilité requise pour tester différents scénarios selon les besoins des utilisateurs.

4. Organisation

L'organisation du projet repose sur une structuration claire des tâches et des équipes afin de garantir une réalisation efficace et coordonnée. Cette section détaille la décomposition en tâches ainsi que la structure des équipes impliquées dans le projet.

4.1 Décomposition en tâches

Le projet est découpé en plusieurs phases, chacune comprenant des tâches spécifiques. Voici la décomposition générale :

1. Analyse des besoins :

- Identification des contraintes fortes et faibles.

- Collecter les attentes des utilisateurs finaux et des parties prenantes.

2. Développement :
- Développement de l'architecture de la base de données.
- Création des interfaces utilisateurs (gestion des contraintes, visualisation).
- Implémentation des fonctionnalités principales (solveur, gestion des données).
- Intégration du solveur pour la génération et l'amélioration de la productivité.

3. Tests et validation :
- Tests fonctionnels, d'intégration et de performance.
- Tests utilisateurs pour évaluer l'ergonomie et la pertinence des solutions générées.

4. Livraison :
- Préparation de l'exécutable final et des fichiers de configuration.
- Rédaction de la documentation utilisateur et technique.

4.2 Structure des équipes

Pour mener à bien ce projet, voici les rôles et responsabilités que nous avons définis :

- Équipe de développement : Responsables de la conception, du codage, et de l'intégration des différentes fonctionnalités, représentée par les étudiants.
- Encadrants académiques : Supervisent le projet, valident les étapes clés et orientent les choix techniques.
- Utilisateur clé : Fournit les données nécessaires aux tests, valide les solutions proposées et participe aux sessions de tests utilisateur, représenté par DPC.

Cette organisation hiérarchique et fonctionnelle permet une répartition claire des responsabilités et facilite la collaboration entre les différentes parties prenantes.

5. Planification

Le projet est organisé en quatres parties, certaines sont dépendantes les unes des autres et d'autres peuvent être réalisées en parallèle dans cette partie nous allons détailler les différentes phrases, puis nous verrons le Gantt relatif au second semestre de notre année scolaire.

5.1 Organisation et préparation des outils collaboratifs

Cette phrase est primordiale, sa durée est fixée à tout le long du projet car en cas de retard, de problèmes, de difficultés, etc. nous devons être en capacité de répondre vite à notre problème avec une nouvelle organisation de travail.

- **Mise à jour de la définition de l'organisation du projet.**
- **Mise à jour des outils existants** :
    - Mise à jour du diagramme de Gantt pour intégrer les nouvelles échéances et assignations.
    - Mise à jour du dépôt GitHub et du Google Drive pour assurer une centralisation des documents (spécifications, diagrammes).

5.2 Développement de la base de données

- **Poursuite de la montée en compétence en la base de données.**
- **Conception de la base de données** :
    - Définition de la structure de la base (base de données relationnelle : entités, relations, contraintes).
    - Création et mise en œuvre de la base initiale.
    - Ajustements et évolutions en fonction des tests de performance ou de structure.
- **Tests de la base de données** :
    - Rédaction de scénarios de test
    - Import de jeux de données pour valider la robustesse et la fiabilité de la base.
    - Documentation des résultats pour prévoir les optimisations nécessaires.

5.3 Intégration du solver

- **Poursuite de la montée en compétence sur le solver**

- **Adaptation du solver aux contraintes spécifiques** :
    - Traduction des contraintes du travail humain en un format exploitable par le solver (contraintes fortes, puis faibles).

- Ajustements et évolutions en fonction des tests de performance ou de structure.

- **Tests du solver :**
  - Définition des scénarios de tests, réalisation des tests et documentation de ses derniers.

5.4 Tests globaux et finalisation

- **Intégration finale** :
  - Validation de l'interaction entre la base de données et le solver.
  - Test des fonctionnalités principales
- **Documentation finale** :
  - Création de manuels utilisateurs et documents techniques pour faciliter la maintenance et la prise en main.
  - Réalisation du rapport du projet et de la présentation pour la soutenance.

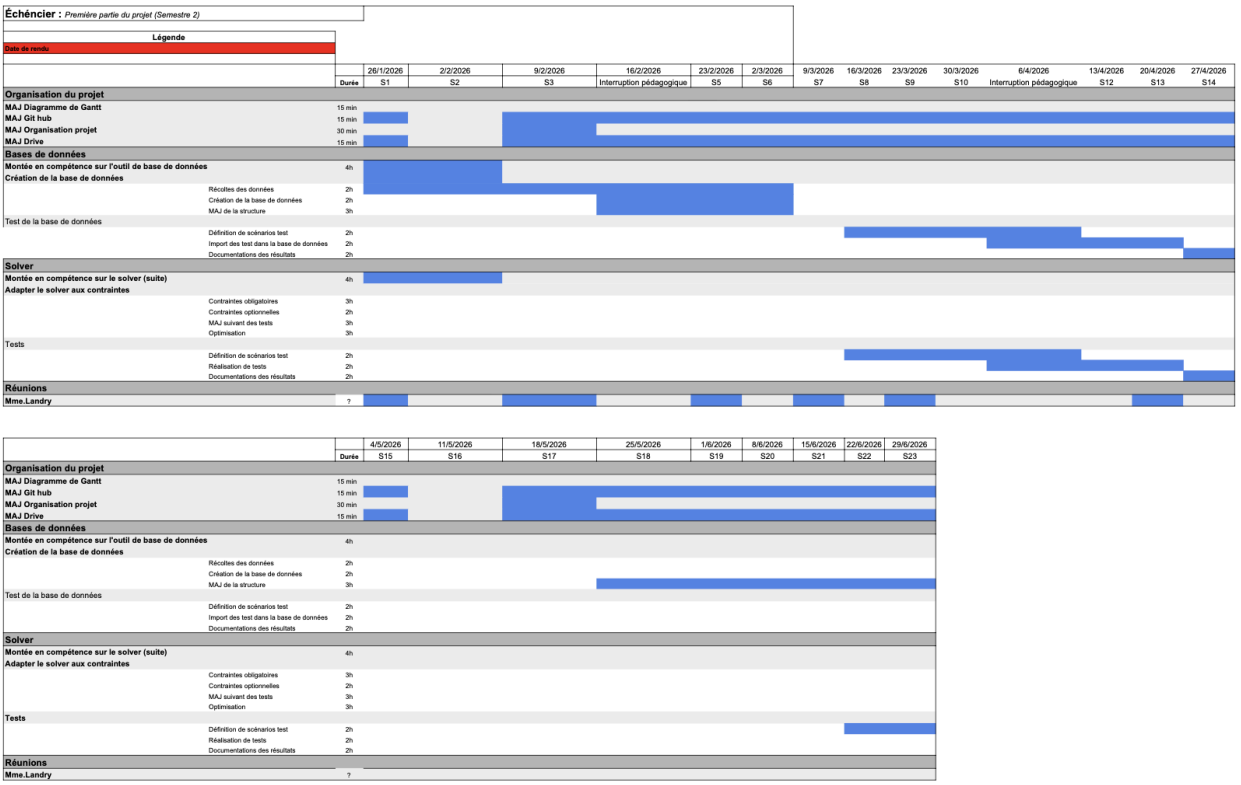
5.5 Calendrier prévisionnel des parties

1. **Janvier 2026 :**
   - Organisation du projet : mise à jour des outils collaboratifs.
   - Poursuite de la montée en compétence sur la base de données (et le solver). 2.

Février 2026 :
   - Définition et création de la base de données.
   - Premiers tests de la base.

3. **Mars 2026 :**
   - Finalisation des tests de la base de données.
   - Documentation des résultats et ajustements nécessaires.
4. **Avril 2026 :**
   - Première phase de développement du solver.
   - Première phase de création d'interfaces.
5. **Mai 2026 :**
   - Intégration complète du solver aux contraintes.
   - Tests globaux et corrections.
   - Poursuite de la création des interfaces et tests d'ergonomie.
6. **Juin 2026 :**
   - Finalisation du projet, tests d'ensemble, et livraison.
   - Documentation utilisateur et technique.

5.6 Représentation sous forme de diagramme de Gantt



6. Cycle de vie

Le cycle de vie du projet suit un modèle structuré en plusieurs phases afin de garantir une progression méthodique et maîtrisée. Chaque phase est conçue pour répondre à des objectifs spécifiques tout en assurant une validation continue des livrables. Ces phases s'appuient sur des méthodes de gestion de projets que nous avons adapté à notre projet, son organisation et disponibilités vis-à-vis de nos cours.

6.1 Phases du Cycle de Vie

1. **Phase d'initiation :**
    - Identification des parties prenantes et définition des objectifs globaux du projet.
    - Élaboration des documents de base, tels que le cahier des charges et le cahier de recette.
2. **Phase de conception :**
    - Création de l'architecture logicielle, y compris la base de données
    - Sélection des outils et technologies
3. **Phase de développement :**

- Implémentation des fonctionnalités principales, comme la gestion des contraintes.
- Intégration des modules (solveur, base de données)..

4. **Phase de tests et validation :**
    - Réalisation de tests unitaires, d'intégration, et fonctionnels.
    - Validation des résultats avec les utilisateurs tests.
5. **Phase de livraison :**

6. **Phase de maintenance et évolutions :**

6.2 Itérations et Validation

Le cycle de vie intègre une approche itérative, permettant des livraisons progressives et des ajustements constants en fonction des retours des parties prenantes. Chaque phase se termine par une validation formelle des livrables, garantissant la continuité et la qualité du projet.

7. Méthodes et outils

Le projet s'appuie sur une méthodologie structurée et des outils performants pour garantir une exécution alignée sur les objectifs fixés. Cette section décrit les méthodes adoptées ainsi que les outils utilisés tout au long du cycle de vie du projet.

7.1 Méthodes

1. **Méthodologie Agile :**
    - Utilisation d'une approche itérative pour permettre des livraisons progressives et des ajustements constants en fonction des retours des utilisateurs.
    - Organisation en sprints définis par des étapes clés comme l'analyse, la conception, le développement et les tests.
2. **Développement orienté sur les besoins :**
    - Mise en place de cycles d'analyse et de validation réguliers avec les parties prenantes pour s'assurer que les fonctionnalités développées répondent aux attentes exprimées.
    - Focus sur la gestion des contraintes (fortes et faibles) pour maximiser la pertinence des solutions générées.
3. **Gestion de projet :**
    - Utilisation d'un planning détaillé (incluant un diagramme de Gantt) pour suivre les délais et répartir les responsabilités.
        - Suivi des livrables à travers des jalons définis pour chaque phase.

7.2 Outils

- **Outils de développement :**
  - LPG : Solveur utilisé pour gérer les contraintes complexes.
  - Environnement de développement intégré (IDE) : Visual Studio Code ou un équivalent pour le codage et la gestion du projet.
- **Outils de gestion de version :**
  - GitHub : Plateforme utilisée pour stocker le code source, collaborer entre membres de l'équipe, et gérer les versions du logiciel.
- **Base de données :**
  - Mise en place d'une base de données structurée pour stocker les informations nécessaires, comme les contraintes, les ressources, et les scénarios.
- **Outils de documentation :**
  - **Microsoft Word / Google Doc** : Pour la rédaction de la documentation technique et utilisateur.
  - **Outils de gestion de planning** : Diagramme de Grantt pour organiser et suivre les tâches du projet.

8. Glossaire

Agile : Méthodologie de gestion de projet favorisant des cycles itératifs et des ajustements continus en fonction des retours utilisateurs.

Base de données relationnelle : Système structuré pour stocker et gérer des données, organisé en tables avec relations et contraintes.

Cahier des charges : Document détaillant les exigences fonctionnelles et techniques du projet.

Cahier de recette : Document définissant les scénarios de tests, les critères de validation et les modalités de vérification du projet.

Contraintes fortes : Conditions indispensables devant être respectées, telles que les indisponibilités des enseignants et la capacité des salles.

Contraintes faibles : Préférences ou optimisations souhaitées, comme la réduction des heures de permanence des élèves.

Diagramme de Gantt : Représentation graphique d'un calendrier de projet, montrant les différentes tâches et leur chronologie.

LPG: Solveur open source développé par Google, utilisé pour résoudre des problèmes complexes sous contraintes.

RGPD : Règlement Général sur la Protection des Données, encadrant la collecte, le traitement et le stockage des données personnelles.

Solveur : Algorithme conçu pour résoudre des problèmes complexes, comme la planification sous contraintes.

Sprint : Cycle court dans une méthodologie Agile, avec des objectifs spécifiques pour chaque phase de développement.

TER (Travail Encadré de Recherche) : Projet académique réalisé dans un cadre universitaire, souvent à titre exploratoire ou pratique.

9. Références

Les références bibliographiques apportant des informations complémentaires :

- Cahier des charges :
    - Version 2.02, datée du 12 janvier 2026.
    - Document définissant les besoins fonctionnels, les contraintes techniques, et les objectifs du projet.

- Cahier de recette :

○ Version 1.02, datée du 12 janvier 2025.
○ Documents contenant les scénarios de tests, les critères de validation, et les modalités de vérification de la qualité du logiciel.

- Plan de Développement :

Version 1.01, datée du 12 janvier 2025.
○ Document décrivant l'organisation, les méthodes, et les outils utilisés pour la réalisation du projet.

- LPG : Spécifications et détails d'implémentation du solveur utilisé.
- PDDL4J's documentation! : Référentiel technique pour le solveur ILog CPLEX.

- Liens vers les outils collaboratifs du projet :

○ TER Andrea BEHAR & Anesie MARTINIANI : Documents centralisés du projet (Google Drive).
○ GitHub: Dépôt pour le code source et les contributions.