

# MiniShell

# Louis Richards

Lilian Rouquette

# Pierre Lacoste

# Introduction

Dans ce document, nous aborderons divers aspects liés au projet de développement d'un système d'exploitation miniature dans le cadre du cours « NSY - Linux ».

Au fil de ce document, nous procéderons à une analyse approfondie des exigences, puis nous détaillerons la conception et le développement du système envisagé pour répondre à ces besoins. Nous fournirons également une description détaillée des fonctions générales du système, mettant en lumière les solutions que nous avons sélectionnées pour sa mise en œuvre.

## Analyse

## Exigences

Le projet exige la réalisation d'un interpréteur basique, similaire à un mini bash, afin de fournir une interface utilisateur conviviale. Parallèlement, une structure interne des inodes, des blocs, et l'arborescence des fichiers doivent être créées, formant ainsi le système de gestion de fichiers (SGF). Un ensemble de fonctions sera défini pour permettre une interaction efficace avec le SGF, facilitant la manipulation des fichiers et des répertoires. Enfin, un ensemble de commandes shell sera développé, exploitant ces fonctions pour offrir des fonctionnalités telles que la navigation dans le système de fichiers, la création de fichiers, et l'exécution de commandes. L'objectif global est de créer un mini-shell complet, capable d'exécuter des commandes de base tout en gérant la structure interne des fichiers de manière efficace.

## Difficultés rencontrés

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aequale doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquid aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguere possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae sine metu degendae praesidia firmissima. – Filium morte multavit. – Si sine causa, nollem me ab eo delectari, quod ista Platonis, Aristoteli, Theophrasti orationis ornamenta neglexerit. Nam illud quidem physici, credere aliquid esse minimum, quod profecto numquam putavisset, si a Polyaeo, familiari suo, geometrica discere maluisset quam illum etiam ipsum dedocere. Sol Democrito magnus videtur, quippe homini erudito in geometriaque perfecto, huic pedalis fortasse; tantum enim esse omnino in nostris.

# Architecture

## Système global

Afin de répondre au mieux aux exigences imposés nous avons décidé de prendre l'architecture donnée comme base.

Bien que le schéma initial nous ait fourni les grandes lignes directrices pour notre projet de MiniOS, nous avons pris l'initiative d'apporter certaines modifications que nous considérons comme nécessaires pour rendre la conception plus logique et modulaire. Les ajustements les plus significatifs ont été réalisés au niveau des responsabilités de l'écran. Dans le schéma initial, l'écran se contentait d'afficher les résultats du shell, tandis que l'utilisateur interagissait avec le processus « Shell » via un terminal. Notre modification fondamentale a consisté à conférer à l'écran la responsabilité de la communication avec l'utilisateur, faisant ainsi de lui l'interface homme-machine (IHM) du projet. Désormais, l'utilisateur saisira ses commandes directement via le processus « Écran », qui se chargera ensuite de transmettre ces commandes au processus « Shell ». Cette adaptation vise à simplifier et rationaliser les échanges entre l'utilisateur et le système, tout en renforçant la modularité de l'architecture globale.

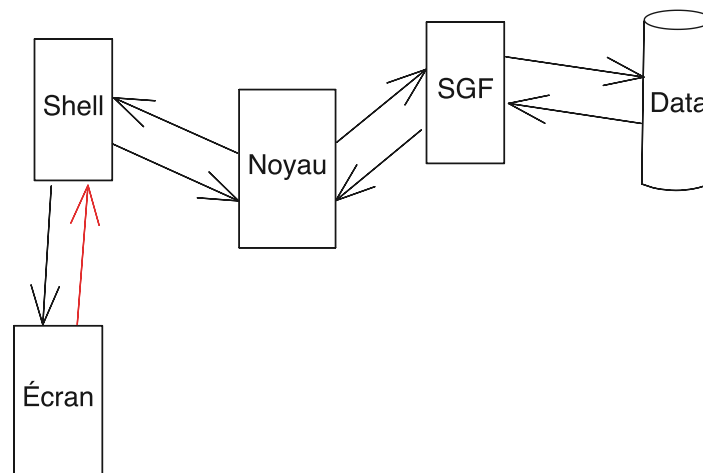


Figure 1 : Architecture revisité

## Fonctions globales

## Procédure de réalisation

## Structure de données

Pour répondre au besoin de simuler un système de fichiers comme celui de Linux, nous avons tout d'abord dû étudier le concept même de ce dernier. Nous avons donc un système avec des « blocks » et des « inodes ».

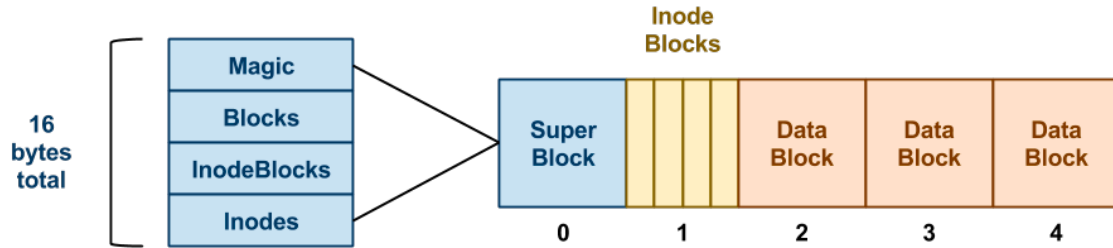


Figure 2 : Architecture blocks & inodes

## Organisation

Pour ce projet, nous étions divisé en groupe de trois étudiants, notre groupe étant constitué de Pierre Lacoste, Lilian Rouquette et Louis Richards. Bien que le groupe soit petit, cela a facilité la communication et le travail collaboratif car nous avons également pu choisir nos groupes ce qui a fait que l'entente au sein de l'équipe était valorisé. Afin de répartir les tâches nous avons mit en place un réunion hebdomadaire tous les [insérer un jour random]. Durant ces réunions nous avons pu discuter de nos avancés, du reste à faire et de se répartir les tâches restantes.

### Outils utilisés

Durant ce projet nous avons utilisé plusieurs outils. Pour l'organisation des tâches nous avons utilisé « ClickUp », pour le gestion de configuration nous avons utilisé « Github », pour la communication nous avons utilisé « Discord ». Ensuite durant le développement nous avons fait recours à différents outils pour nous aider tel que « Valgrind », « CppCheck », « Gdb » et « lldb ».

## Conclusion

### Lancer le projet

*(Toutes les commandes sont à executer à la racine du projet)*

Pour lancer le projet suivez les étapes suivantes :

1. Télécharger le code source depuis [github.com/LouisRichards/MiniShell/tree/main](https://github.com/LouisRichards/MiniShell/tree/main)
2. Dans votre terminal, executer la commande « make ».
3. Toujours dans votre terminal, executer la commande « ./bin/screen ».

Pour lancer les tests de ce projet suivez les étapes suivantes :

1. Dans votre terminal, executer la commande « make tests ».
2. Vous devriez observer les résultats des tests dans votre terminal.