RETOUR D'EXPÉRIENCE

Projet LP25 – Moniteur de processus Linux

ISLAM Abir • MELLOUK Mohamed-Amine • FALLANI Issam

Université de Technologie de Belfort-Montbéliard (UTBM)

Janvier 2026

Table des matières

RETOUR D'EXPÉRIENCE

Projet LP25 – Moniteur de processus Linux

Auteurs : ISLAM Abir, MELLOUK Mohamed-Amine, FALLANI Issam
Date : Janvier 2026

---

1. Introduction

Le projet LP25 consistait à développer un outil de gestion et de surveillance de processus sous Linux, similaire à l'utilitaire **htop**. L'application devait proposer deux modes de fonctionnement distincts : - **Mode local** : surveillance des processus sur la machine hôte - **Mode réseau** : surveillance simultanée de plusieurs machines distantes via SSH

Ce document présente un retour d'expérience sur les difficultés rencontrées, les solutions apportées, ainsi qu'une analyse critique de notre méthodologie de travail.

---

2. Contexte et objectifs du projet

2.1 Objectifs fonctionnels

L'objectif principal était de créer un moniteur de processus offrant les fonctionnalités suivantes : - Affichage en temps réel de la liste des processus avec leurs caractéristiques (PID, nom, CPU, mémoire, etc.) - Interface utilisateur interactive basée sur la bibliothèque **ncurses** - Gestion des processus : pause (SIGSTOP), reprise (SIGCONT), arrêt (SIGTERM), et terminaison forcée (SIGKILL) - Recherche et filtrage de processus - Support multi-machines avec navigation par onglets en mode réseau

2.2 Contraintes techniques

Le projet imposait plusieurs contraintes techniques : - Langage de programmation : **C** - Utilisation de la bibliothèque **ncurses** pour l'interface en terminal - Accès au système de fichiers **/proc** pour récupérer les informations des processus - Utilisation de la bibliothèque **libssh** pour les connexions distantes - Gestion sécurisée des identifiants (fichier de configuration avec permissions 600)

---

3. Difficultés rencontrées

3.1 Phase d'analyse et de compréhension

Démarrage difficile : Dès la découverte du sujet, le projet nous est apparu comme extrêmement complexe. La première difficulté a été de comprendre précisément les attentes et de visualiser une architecture logicielle claire.

Manque de planification initiale : Nous avons souffert d'un manque de préparation dans la phase d'analyse. Une meilleure décomposition du projet en sous-modules dès le début aurait grandement facilité le développement.

3.2 Prise en main de ncurses

La bibliothèque **ncurses** était totalement nouvelle pour nous. Ses particularités ont nécessité un apprentissage important : - Gestion du rafraîchissement de l'affichage - Manipulation des fenêtres (windows) et des sous-fenêtres - Gestion des entrées clavier sans bloquer l'affichage - Contrôle des couleurs et de la mise en forme

Solution adoptée : Nous avons dû consulter intensivement la documentation officielle et réaliser de nombreux tests pour maîtriser les concepts de base (fenêtres, pads, gestion du clavier, etc.).

3.3 Accès et analyse du système /proc

La lecture et l'interprétation des fichiers dans `/proc` ont également posé des défis : - Parsing des fichiers `/proc/[pid]/stat` et `/proc/[pid]/status` - Calcul du pourcentage d'utilisation CPU - Gestion des processus qui apparaissent/disparaissent rapidement - Gestion des permissions (certains processus nécessitent les droits root)

3.4 Mode réseau et SSH

Le mode réseau a représenté la partie la plus complexe du projet.

Problématiques rencontrées : - Compréhension de l'architecture client-serveur via SSH - Installation et configuration des serveurs SSH de test - Gestion de l'authentification (mot de passe vs clé SSH) - Exécution de commandes distantes et récupération des résultats - Gestion des erreurs de connexion et des timeouts - Affichage multi-onglets pour plusieurs machines

Solution adoptée : Nous avons choisi d'utiliser la bibliothèque **libssh** qui offre une API C pour établir des connexions SSH. Nous avons mis en place : - Une structure de configuration permettant de définir plusieurs hôtes distants - Un système de connexion avec authentification par mot de passe - Une commande distante qui exécute des commandes shell pour récupérer les informations des processus - Un système d'onglets (tabs) dans l'interface ncurses pour naviguer entre les machines

3.5 Gestion des interactions clavier

L'implémentation des raccourcis clavier pour tuer un processus, le mettre en pause ou le rechercher a nécessité une gestion fine des événements : - Capture des touches sans bloquer l'affichage - Validation des actions critiques (confirmation avant kill) - Gestion de la saisie de texte pour la recherche - Compatibilité avec différentes configurations de terminal

---

4. Solutions techniques mises en œuvre

4.1 Architecture logicielle

Le projet a été structuré en modules distincts pour faciliter la maintenance :

```
src/
├── main.c       - Point d'entrée et parsing des arguments
├── manager.c/h  - Orchestration multi-machines
├── process.c/h  - Gestion des processus Linux (/proc)
├── network.c/h  - Connexions SSH et hôtes distants
└── ui.c/h       - Interface ncurses avec système d'onglets
```

Cette séparation a permis de travailler en parallèle sur différentes fonctionnalités.

4.2 Gestion de la mémoire

Nous avons mis en place une gestion rigoureuse de la mémoire pour éviter les fuites : - Libération systématique des structures de données - Fermeture des connexions SSH en cas d'erreur - Nettoyage de ncurses avant la sortie du programme

4.3 Gestion des erreurs

Chaque module intègre une gestion d'erreurs robuste : - Vérification des permissions pour les opérations sensibles - Messages d'erreur clairs pour l'utilisateur - Mode `--dry-run` pour tester l'accès aux processus sans interface

4.4 Sécurité

Nous avons accordé une attention particulière à la sécurité : - Fichier de configuration avec permissions obligatoires (chmod 600) - Pas de mots de passe en ligne de commande (visible dans l'historique) - Validation des entrées utilisateur

---

5. Auto-critique et axes d'amélioration

5.1 Points faibles identifiés

Manque de planification initiale : Avec du recul, nous aurions dû consacrer plus de temps à l'analyse du sujet dès le début. Une phase de conception approfondie (diagrammes UML, architecture détaillée) aurait permis de mieux anticiper les difficultés.

Apprentissage tardif des outils : Nous avons découvert ncurses et libssh en cours de développement. Une formation préalable ou une phase de prototypage aurait réduit le temps perdu en erreurs et tests.

Gestion du temps : La complexité du mode réseau a monopolisé beaucoup de temps, au détriment de certaines fonctionnalités secondaires (tri, filtres avancés).

Communication d'équipe : Nous aurions pu améliorer la communication au sein de l'équipe, notamment pour partager plus rapidement les solutions trouvées individuellement.

5.2 Points forts

Persévérance : Malgré les difficultés, nous avons réussi à mettre en œuvre les fonctionnalités principales.

Modularité : La structure du code favorise la réutilisabilité et faciliterait l'ajout de nouvelles fonctionnalités.

Robustesse : Le programme gère correctement les erreurs et les cas limites.

5.3 Améliorations possibles

Si nous devions poursuivre le projet, voici les améliorations que nous proposerions :

Fonctionnalités supplémentaires : - Tri des processus par différents critères (CPU, mémoire, nom, etc.) - Graphiques d'utilisation des ressources (CPU, mémoire) dans le temps - Support des clés SSH pour l'authentification - Export des données en CSV ou JSON - Mode démon pour surveillance continue

Optimisations techniques : - Utilisation de threads pour améliorer les performances en mode réseau - Cache des connexions SSH pour réduire la latence - Compression des données transmises - Support du protocole IPv6

Interface utilisateur : - Personnalisation des couleurs - Sauvegarde des préférences utilisateur - Aide contextuelle plus détaillée

---

6. Compétences acquises

Ce projet a été extrêmement formateur et nous a permis de développer de nombreuses compétences :

6.1 Compétences techniques

- **Programmation système Linux** : manipulation de `/proc`, gestion des processus, signaux UNIX
- **Programmation en C** : gestion mémoire, pointeurs, structures de données
- **Bibliothèque ncurses** : création d'interfaces textuelles interactives
- **Protocole SSH** : connexions distantes, authentification, exécution de commandes

- **Débogage** : utilisation de gdb, valgrind pour détecter les fuites mémoire

6.2 Compétences méthodologiques

- **Architecture logicielle** : conception modulaire, séparation des responsabilités
- **Gestion de projet** : planification, répartition des tâches
- **Documentation** : rédaction de README, commentaires de code
- **Versionning** : utilisation de git pour le travail collaboratif

6.3 Compétences transversales

- **Autonomie** : recherche documentaire, résolution de problèmes complexes
- **Travail d'équipe** : collaboration, communication, partage de connaissances
- **Adaptabilité** : face à des technologies nouvelles et des problèmes imprévus
- **Rigueur** : gestion des erreurs, tests, sécurité

---

7. Conclusion

Le projet LP25 a représenté un défi technique important, mais également une expérience pédagogique très riche. Malgré les difficultés rencontrées, notamment lors de la prise en main de ncurses et de l'implémentation du mode réseau, nous sommes parvenus à développer un outil fonctionnel répondant aux exigences principales.

Ce projet nous a permis de comprendre en profondeur le fonctionnement interne d'un système Linux, de la gestion des processus à la communication réseau. Il a également mis en évidence l'importance d'une phase de conception solide et d'une bonne maîtrise des outils avant de commencer le développement.

Principaux enseignements : - La préparation et la planification sont essentielles pour les projets complexes - La documentation technique est un outil indispensable - L'apprentissage par la pratique, bien que parfois frustrant, est extrêmement efficace - La modularité du code facilite grandement la maintenance et l'évolution

Cette expérience nous sera très utile pour nos futurs projets, tant sur le plan technique que méthodologique. Elle nous a montré que même face à des difficultés importantes, la persévérance et une approche structurée permettent de surmonter les obstacles.

Signatures :

ISLAM Abir
MELLOUK Mohamed-Amine
FALLANI Issam

Date : Janvier 2026