UNIVERSITE DE ROUEN

ARCHITECTURE DU LOGICIEL

Version :	1.1
Date :	< date de publication du document>
Rédigé par :	Emmanuel Mocquet
Relu par :	< nom des membres du projet ayant relu le document>
Approuvé par :	

Objectif : Le but de ce document est de décrire les solutions techniques conçues pour répondre aux exigences définies dans la spécification technique de besoin. Il doit identifier et décrire les différents modules ou constituants du logiciel ainsi que leurs interfaces de telle sorte que chacun d'entre eux puisse être développé de façon autonome par un membre de l'équipe avant d'être intégré.
La lecture de ce document doit également permettre d'appréhender l'ensemble des paramètres techniques pris en considération par les auteurs pour définir et élaborer les stratégies et démarches de développement.

MISES A JOUR

Version	Date	Modifications réalisées
0.1	16/12/11	Création
0.2	10/01/12	Complétion et modifications
1	16/01/12	Suppression d'un module inutile
1.1	14/03/12	Ajustements par rapport à la STB

1. Objet :

Ce document a pour but de décrire les solutions techniques conçues pour répondre aux exigences définies dans la spécification technique de besoin. Il doit identifier et décrire les différents modules ou constituants du logiciel ainsi que leurs interfaces de telle sorte que chacun d'entre eux puisse être développé de façon autonome par un membre de l'équipe avant d'être intégré.

Ce logiciel doit simuler un périphérique, composé d'un SSD et d'un disque dur mécanique, et permettre le contrôle par le noyau de ce périphérique.

La lecture de ce document doit également permettre d'appréhender l'ensemble des paramètres techniques pris en considération par les auteurs pour définir et élaborer les stratégies et démarches de développement.

2. Documents applicables et de référence

Le sujet du projet est disponible dans le document « projet1-2-3.pdf ».
Le document de spécification est intitulé PBV_STB_2.

3. Terminologie et sigles utilisés

- Module noyau: Programme qui peut être chargé dynamiquement dans le noyau.
- SSD: Solid State Disk, périphérique de stockage qui utilise de la mémoire flash. Les performances sont élevées et les consommations d'énergie basses mais le coût par Go est élevé.
- HDD: Hard Drive Disk, périphérique de stockage de masse. Il consomme plus et ses performances sont moindres qu'un SSD mais le prix d'acquisition est beaucoup moins élevé.
- PBV: Périphérique Bloc Virtuel. Il s'agira ici d'un fichier spécial de type bloc représentant deux disques (SSD et HDD). Il est cependant possible qu'il ne soit composé que du HDD.
- SGF: Un Système de Gestion de Fichiers « est une façon de stocker les informations et de les organiser dans des fichiers ». (Wikipedia)
- Pilote: Programme gérant les opérations entre un périphérique et le reste du système.
- Mémoire cache: Mémoire non volatile d'accès rapide. Les données ne sont pas liées à un processus en cours d'exécution.

4. Architecture physique du matériel utilisé

Le but de ce projet étant d'implanter une optimisation de fonctionnement de SSD et HDD simultanée, tout processeur, même de faible puissance, est suffisant.

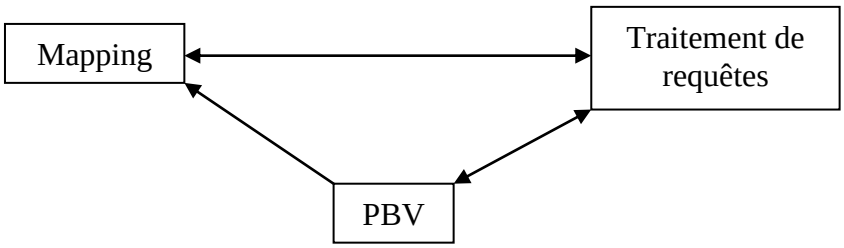
Ces deux disques ne sont indispensables qu'en phase avancée du projet : il est possible de travailler avec des disques virtuels. Cela réduira en autres le nombre d'écritures et lectures sur le SSD.

Le logiciel devra être opérationnel sur un OS de type Linux, dont la version du noyau sera au moins la 2.6.

5. Architecture statique du logiciel

5.1. Structure logique

Le schéma ci-dessous représente la structure logique de la solution que nous proposons. Un second schéma, plus global et représentatif de la problématique, est disponible en annexe. Il schématise le projet dans son ensemble, et l'architecture ici spécifiée.



5.2. Description des constituants

5.2.1. Description du constituant « Mapping »

- Son rôle sera de permettre une correspondance entre les blocs du disque dur mécanique et du SSD ;
- ce module contiendra des fonctions capables de fournir des blocs pour un paramètre donné (comme par exemple, un mode d'accès ou un identifiant disque) ;
- ce constituant communiquera avec celui intitulé « Traitement de requêtes » décrit plus loin. Il recevra en outre des informations de la part du module « PBV » qui lui indiquera les disques dont il faudra « mapper » les blocs ;
- sa taille sera petite, tout comme sa complexité.

5.2.2. Description du constituant « Traitement de requêtes »

- Son rôle sera traiter les requêtes de lecture et d'écriture ;
- il communiquera avec les constituants « Mapping » et « PBV ». Le premier fournira les adresses auxquelles devront s'effectuer la lecture et l'écriture, tandis que le second permettra de connaître les disques avec lesquels il faut communiquer ;
- sa taille et sa complexité seront petits.

5.2.3. Description du constituant « PBV »

- Son rôle sera de contenir toutes les informations propres aux disques que nous allons « fusionner » ;
- ses attributs seront, entre autres, des pointeurs sur des disques, une file de requêtes…
- il communiquera avec les constituants précédents pour les raisons évoquées ci-dessus ;
- sa taille et sa complexité sont très petits.

6. Justifications techniques

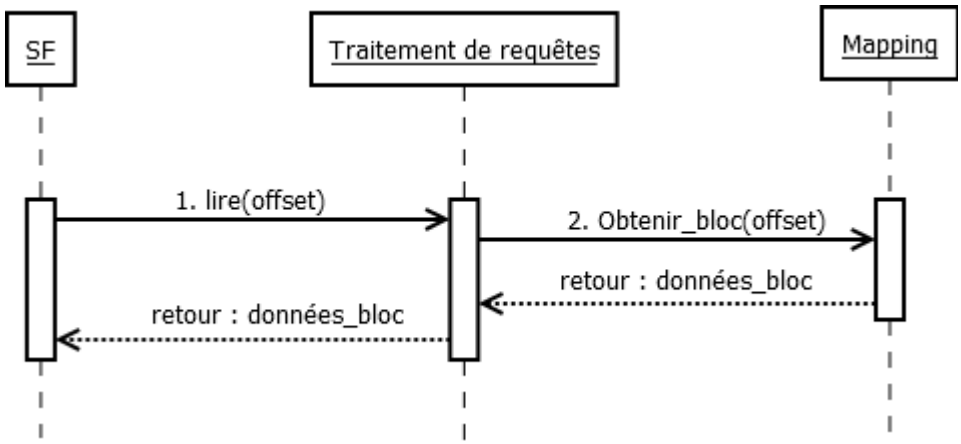
Noyau : nous avons posé une condition sur la version du noyau à utiliser. En effet, certaines fonctions nécessaires dans le traitement des requêtes voient leur spécification changer (dont le type de retour).

Langage : nous travaillerons en C puisqu'il est un langage spécifique à la programmation système. Il nous permettra de développer des modules noyau et ainsi un pilote. C'est aussi le langage dans lequel nous sommes les plus à l'aise.

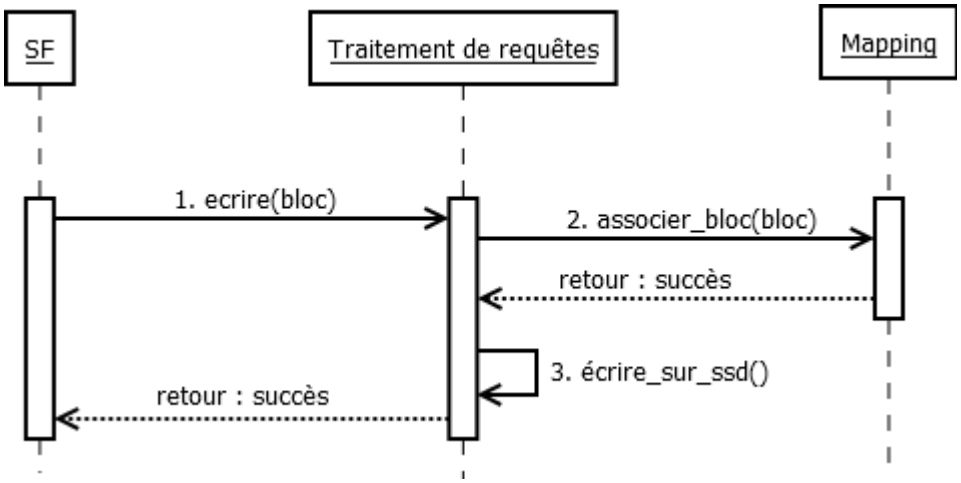
Périphérique Bloc Virtuel : cette approche, par opposition à la réimplantation d'un système de fichiers, nous permet une plus grande flexibilité. La simulation d'un tel périphérique offre la possibilité de s'affranchir du système de fichiers et de garantir ainsi une meilleure portabilité.

7. Fonctionnement dynamique

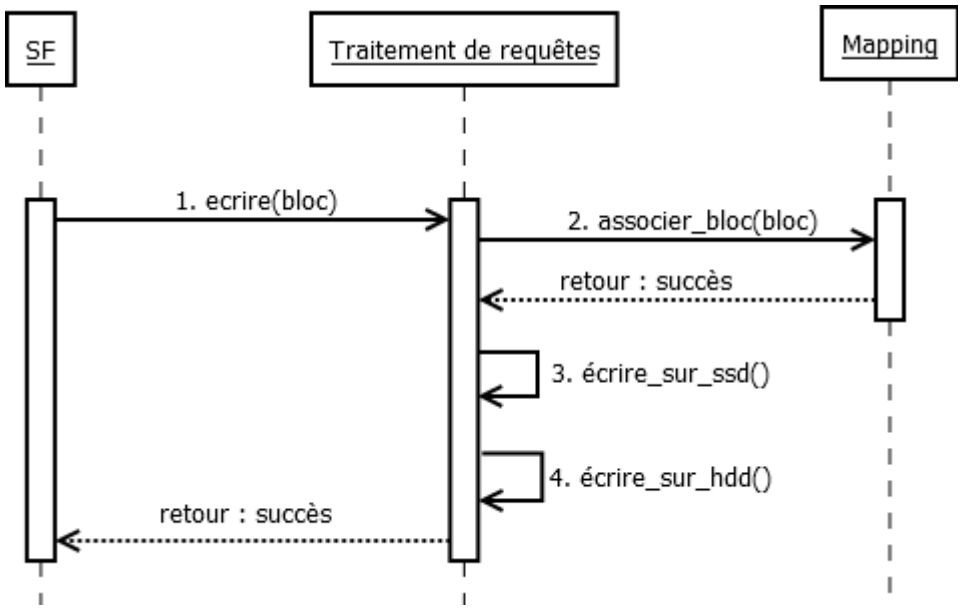
- Lecture de bloc (flot principal) :
  Composants mis en jeu : Système de fichiers (SF), Traitement de requêtes et Mapping.
  Une précondition est l'enregistrement préalable du PBV.



- Ecriture de bloc en mode économie (flot principal) :
  Composants mis en jeu : Système de fichiers (SF), Traitement de requêtes et Mapping.
  Une précondition est l'enregistrement préalable du PBV.



- Ecriture de bloc en mode sécurité (flot principal) :
  Composants mis en jeu : Système de fichiers (SF), Traitement de requêtes et Mapping.
  Une précondition est l'enregistrement préalable du PBV.



8. Traçabilité

Se reporter à la spécification technique de besoin (STB) pour l'association des numéros de fonctions et leur description.

Les exigences F-G1-10, F-G1-20 et F-G1-30 ne sont pas satisfaites par les composants décrits ici. Ils le seront cependant par la fonction d'initialisation du pilote. C'est parce que nous considérons que cette fonction n'est pas un composant que nous ne l'avons pas spécifiée précédemment.

Exigences		F-G1-10	F-G1-20	F-G1-30	F-G1-40	F-G1-50	F-G1-60	F-G1-70
Composants	Mapping			x	x			
	Traitement de requêtes			x	x			
	PBV			x	x			

9. Annexe

Schéma du fonctionnement global du pilote et des interactions avec les périphériques et le noyau.