

PROJET SYSTÈME D' EXPLOITATION / UNIX / LINUX

M1 Master MIAGe

Le planning + Les sujets + Les recommandations pour la rédaction du dossier de projet

Planning projet SE 2017/18

06/12: Présentation des sujets

11/12 au plus tard : envoyer par mail la constitution des groupes de 3 étudiants (liste des noms) + classement des 2 sujets

12/12 : attribution des sujets aux groupes

15/01 **au plus tard**: remettre les dossiers sous l'ENT

17/01: Démonstrations

SUJET 1

SERVEUR de FICHIER PARTAGÉ

Il s'agit de régler l'accès aux enregistrements d'un fichier utilisé simultanément par plusieurs programmes. Pour cela on mettra en œuvre le modèle client/serveur qui permet une plus grande sécurité que l'utilisation des verrous.

Le programme **session** permet de faire une succession de requêtes de six types :

- C pour la consultation d'un enregistrement
- M pour la modification d'un enregistrement
- S pour la suppression d'un enregistrement
- A pour l'adjonction d'un enregistrement
- V pour la visualisation du fichier complet
- F pour indiquer la fin des requêtes

Ce programme peut être lancé par plusieurs utilisateurs simultanément selon la syntaxe :

syntaxe : session [nom de fichier]

Un serveur est chargé d'effectuer les accès aux enregistrements afin de régler les accès concurrents lors de modification ou de suppression d'un même enregistrement. Il est lancé en arrière-plan (démon) par la commande **servfic** :

syntaxe : servfic -f <nom de fichier> [-d <nbre de secondes>]

La modification d'un enregistrement sera toujours constituée de 4 opérations :

- * lecture par le serveur
- * affichage du contenu courant de l'enregistrement à modifier
- * modification de ce contenu de façon interactive
- * écriture du nouveau contenu par le serveur

La suppression sera toujours constituée de :

- * lecture par le serveur
- * affichage du contenu
- * demande de validation
- * suppression effective par le serveur ou annulation de la demande de suppression

Les validations de suppression et les modifications à effectuer doivent avoir lieu dans un délai raisonnable fixé à l'appel du serveur par l'option **-d** afin d'éviter des blocages (fichier verrouillé indéfiniment par le serveur). Rappel : une suppression ne peut être effectuée que si aucune autre requête n'est en cours d'exécution. De même pour la modification.

Les requêtes de suppression et de modification doivent être traitées en priorité.

Les utilisateurs effectuant simultanément des requêtes de consultation, visualisation ou adjonction ne doivent pas se bloquer entre eux.

Prévoyez une démonstration "parlante".

SUJET 2

GESTION de PROCESSUS en BATCH

La gestion de processus en batch permet à tout utilisateur de lancer de façon cyclique à une date ou une heure précise un programme. Par exemple, lancement automatique du programme PAIE tous les 26 de chaque mois à 23h30, ou lancement automatique d'un programme de sauvegarde du contenu d'un disque tous les lundis à 3h du matin.

La mise en œuvre d'un tel système nécessite l'écriture d'un programme démon **gobatch** qui explore un fichier **fbatch** dans lequel sont définis les programmes ou commandes à exécuter cycliquement.

L'utilisateur qui veut lancer un programme cyclique ou le supprimer doit lancer la commande **pgcycl** en précisant au besoin des paramètres comme :

- 1 minute 0-59
- 2 heure 0-23
- 3 jour du mois 1-31
- 4 mois 1-12
- 5 jour de la semaine 0-6
- 6 commande à exécuter en précisant le nom du fichier de sortie et le nom du fichier des erreurs

pgcycl modifie en conséquence le fichier fbatch.

La sortie standard (stdout) et la sortie des erreurs (stderr) des commandes soumises à gobatch sont redirigées vers les fichiers précisés dans le champ 6.

Le démon gobatch, lancé en arrière-plan, est averti à chaque modification du fichier par pgcycl.

Par souci de simplification, gobatch ne travaille que pour le compte d'un utilisateur et est lancé une fois pour toutes.

syntaxe de pgcycl :

pgcycl <-l> fbatch *pour lister le contenu de fbatch*
 <-d> fbatch *pour détruire une ligne de fbatch*
 <-a> fbatch *pour ajouter une ligne à fbatch*

Prévoyez une démonstration "parlante".

Conseils pour la rédaction d'un dossier de Projet

surlignés en jaune les éléments très importants dans votre dossier de projet SE

Structure d'un dossier

La réalisation d'un projet a de multiples objectifs : apprendre à résoudre des exemples plus complexes que ceux qui peuvent être abordés en travaux dirigés, approfondir une solution possible, se familiariser avec un langage de programmation, apprendre à tester un programmes mais aussi à présenter le travail réalisé. Ce dernier point est capital et mérite de s'y attarder. En effet, la rédaction d'un dossier, qu'il soit de programmation ou d'autre chose, est un travail difficile et de lui dépendra ...bien évidemment la note dans le cas présent mais de manière plus générale le jugement porté sur l'ensemble du travail. En un mot, c'est le dossier qui fera vendre le programme.

Le programme résout le problème posé. Ses qualités sont sa correction (résout-il effectivement le problème posé ?), sa complétude (le fait-il dans tous les cas de figure possibles ?), son efficacité, sa facilité d'utilisation, mais aussi sa lisibilité (est-il compréhensible par d'autres ?) et sa capacité à évoluer. Certaines de ces qualités peuvent être testées (voire même automatiquement), mais c'est le dossier qui permet de comprendre

- La démarche ou la méthode suivie,
- La solution adoptée (Thread ou processus, file de messages ou mémoire partagée ou sémaphores ou signaux, ...)
- Les choix effectués
- La structure des objets introduits
- Ce que réalisent les fonctions définies ainsi que leur imbrication les unes par rapport aux autres
- Le texte du programme

Et aussi

- De prouver sa correction
- D'avoir une évaluation chiffrée de ses caractéristiques techniques
- De faire évoluer, de réutiliser le programme ou de corriger certaines erreurs qui ne manqueront pas de subsister et enfin
- De l'utiliser correctement

En un mot, le dossier reflète beaucoup de choses, et en particulier la compréhension du problème et le recul pris vis à vis de lui par l'auteur. La structure générale d'un dossier de projet est la suivante :

1. Introduction
2. Dossier Utilisateur
3. Dossier Concepteur
4. Dossier de Programmation
5. Dossier de Tests
6. Conclusion

Introduction

Comme tout rapport, un dossier de projet doit commencer par une introduction. Celle-ci rappelle le problème posé, sans recopier toutefois l'énoncé, en apportant plus de précision et en justifiant les interprétations faites et les grandes limitations fixées dès le départ. Dans certains cas, il faut également préciser l'environnement de développement, la liste des outils fournis, etc. La solution choisie ainsi que le plan du rapport seront ébauchés.

Dossier utilisateur

Contenu et destinataire. Comme son nom l'indique, ce dossier est destiné à un utilisateur potentiel du programme, non nécessairement informaticien et encore moins programmeur. Il doit contenir tous les renseignements nécessaires et suffisants pour une bonne utilisation ainsi que les renseignements concernant les problèmes pouvant surgir. En particulier, c'est ici qu'est décrite l'interface utilisateur.

Structure proposée

1. Appel : répertoire d'accès au programme, nom et profil de la commande à utiliser (c'est-à-dire sens de chaque paramètre, s'il y en a), outils nécessaires à l'utilisation du logiciel
2. Action : le traitement réalisé par la commande, c'est-à-dire les données indispensables, les résultats produits, les erreurs détectées
3. Déroulement de l'exécution : depuis la saisie des données jusqu'à la production des résultats. Il est nécessaire de préciser le type, la précondition (souvent il s'agit d'un intervalle de valeur), le format d'entrée et le lieu de saisie des données sur l'écran. Un exemple d'exécution aide à la compréhension. Dans le cas d'une interface sophistiquée, son utilisation doit être détaillée. En particulier, les différents menus doivent être présentés et des images des écrans de dialogue.
4. Erreurs : la liste de toutes les erreurs d'utilisation détectées par le programme, les messages correspondants ; que faire en cas d'erreur ?

Conseils

- Le dossier utilisateur doit être concis mais complet. Un exemple simple d'utilisation dans un cas normal facilite la compréhension.
- La commande proposée à l'utilisateur doit être accessible à tous et si possible auto-documentée (un appel sans paramètre rend un message d'utilisation). Si elle utilise un ou plusieurs fichiers de données, ceux-ci doivent pouvoir se trouver n'importe où dans l'arborescence des fichiers et des répertoires et pas nécessairement dans le répertoire d'appel. Si la commande crée des fichiers intermédiaires, ceux-ci doivent être supprimés en fin de commande. De façon générale, l'utilisateur doit retrouver son environnement de travail dans le même état qu'avant l'appel de la commande.

Dossier concepteur

Contenu et destinataire. Il est destiné à une personne qui veut comprendre la méthode de conception suivie, le raisonnement appliqué, les choix effectués et les raisons de ces choix. Elle doit y trouver tous les renseignements lui permettant de modifier, corriger, faire évoluer et réutiliser tout ou partie du travail.

Structure proposée. La structure peut être modulée en fonction de la complexité du problème à résoudre et de l'analyse suivie.

1. Analyse : Il est inutile de donner dans ce dossier une paraphrase de l'algorithme. Il faut détailler chaque phase du processus de développement suivi lors de la résolution du problème, en particulier le raisonnement appliqué, **les choix des outils et les raisons des solutions adoptées**. Ainsi, le dossier concepteur final doit être fidèle à la démarche suivie. **L'utilisation de schémas** (montrant par exemple, le « flux » des données, signaux, ... entre les différentes tâches) est souvent une aide efficace pour expliquer votre solution.
2. Algorithmes synthétiques.

Dossier de programmation

Contenu et destinataire. Il est destiné au programmeur qui devrait éventuellement reprendre le code. Il explique le passage de l'analyse du problème au programme.

Le traitement de l'interface utilisateur doit apparaître ici, car il dépend fortement du langage utilisé et de l'environnement de travail dont on dispose.

D'autre part, certaines modifications peuvent être apportées localement à l'analyse pour des raisons d'efficacité. Ces changements ne doivent pas être passés sous silence. Il est cependant important que le dossier concepteur reste cohérent avec le programme.

Structure proposée

1. Passage au langage de programmation : donner la représentation concrète de types d'objets introduits dans l'analyse.
2. Interface : Le traitement de l'interface utilisateur est souvent ignoré dans l'analyse. C'est dans ce dossier qu'il doit apparaître. A ce sujet, ne jamais oublier que le développement d'une interface utilisateur sophistiquée n'est en général pas l'objet du projet.
3. Programmes : Les textes des programmes non présentés dans ce dossier ainsi que les scripts utilisés. Les caractéristiques techniques doivent être citées : nombre de lignes de code, taille de l'exécutable, etc.

Conseils de présentation et de rédaction

- Les programmes doivent être commentés :
 1. Lexique de chaque déclaration
 2. Commentaires notant l'entrée puis la sortie des blocs décrits dans les algorithmes
 3. Eventuellement des remarques dans un corps de fonction
- Une variable locale ne doit pas être utilisée pour deux objets différents
- Les noms doivent être les mêmes dans l'analyse et dans le programme
- Les programmes font partie intégrante de ce dossier : ils doivent être présentés de la même façon (feuilles au format A4).

Dossier de tests

Contenu et destinataire. Bien souvent destiné au même lecteur que le dossier précédent, ce dossier doit néanmoins pouvoir être lu par n'importe qui. Il indique quels sont les tests qui ont été effectués pour garantir au maximum le bon fonctionnement du programme obtenu. Pour savoir quels sont les tests à inclure dans ce dossier, il faut identifier tous les cas possibles qui peuvent se présenter, les cas normaux et les cas d'exceptions.

Structure proposée. Deux structures sont possibles et complémentaires :

- Structure par type et par fonction
- Structure par type de test : cas normaux, limites et d'erreurs

Un test est composé d'un tableau de test et du listing correspondant. Un tableau de test a la structure suivante :

Cas précis testé	Donnée fournie (valeur ou propriété)	Résultat attendu (valeur ou propriété)	Résultat obtenu (si différent de l'attendu)

--	--	--	--

Le dossier de test doit comporter tous les cas à tester. Mais certains (faute de temps, d'espace, etc.) peuvent ne pas avoir été exécutés.

Conclusion

Tout rapport doit se terminer par une conclusion. Elle est destinée au lecteur du dossier (et tout spécialement au correcteur !). Elle doit faire un état des lieux !

Ce que fait effectivement le programme ;

Les erreurs ou incorrections détectées ;

Les cas non traités

Puis une étude prospective :

- Les évolutions possibles du programme ;
- Les parties éventuellement réutilisables

Pour terminer par quelques commentaires personnels sur la réalisation, les problèmes rencontrés, les apports du projet, etc.

Derniers conseils

Un dossier de projet est un document technique mais **il est destiné à être lu par un être humain !** Il est important que le dossier soit complet, mais aussi qu'il soit agréable à lire. D'une manière générale, il faut penser aux lecteurs, leur faciliter les différents types de lecture :

- Lecture exhaustive : ne pas se répéter mais faire des références précises (éviter les références en avant), éviter les romans fleuves et le style télégraphique, faire des phrases courtes, n'écrire que sur le recto des feuilles en laissant une marge tout autour, etc.
- Lecture en diagonale : chaque partie doit être un ensemble bien structuré, agrémenté de graphes, de schémas et d'exemples. La cohérence entre les différentes parties doit être préservée.
- Lecture directe pour aller chercher une information précise : il faut faciliter l'accès rapide (**plan au début, repères en couleur, numérotation des pages**, etc.), présenter un seul dossier dont toutes les pages s'ouvrent dans le même sens.

D'autre part, chaque partie est destinée à un lecteur différent, il faut en tenir compte. Enfin, il ne faut pas oublier que le dossier est le reflet de la bonne ou de la mauvaise compréhension du problème, du recul pris vis à vis du sujet et de la maîtrise de la solution de ses auteurs.