|  |
| --- |
| Λειτουργικά Συστήματα |
| 2η υποχρεωτική εργασία |
| Αναγνώστου Αντώνης – 2268  Γωγούσης Παύλος – 2251  Λασκαρίδης Στέφανος - 2315 |

|  |
| --- |
| Τμήμα Πληροφορικής ΑΠΘ  2-13-2015 |

Περιεχόμενα

[Περιγραφή εργασίας 2](#_Toc411626329)

[Γενικές παραδοχές 2](#_Toc411626330)

[Προβλήματα 2](#_Toc411626331)

[Μελλοντικές επεκτάσεις 2](#_Toc411626332)

[Σύνοψη 3](#_Toc411626333)

# Περιγραφή εργασίας

Η παρούσα εργασία αποτελεί την δεύτερη υποχρεωτική άσκηση για το μάθημα των Λειτουργικών Συστημάτων και αφορά στην κατασκευή ενός κελύφους (shell) για το GNU Linux.

Για την ανάπτυξη χρησιμοποιήθηκε το περιβάλλον Eclipse Luna (for C++), C++11 σε λειτουργικό σύστημα Ubuntu 12.04 και 14.04 LTS 64bit.

Στην τρέχουσα αναφορά μπορείτε να βρείτε τις γενικές παραδοχές και τα προβλήματα που αντιμετωπίσαμε κατά την ανάπτυξη του προγράμμματος, ενώ λεπτομερή περιγραφή των μεθόδων και κλήσεων που χρησιμοποιήθηκαν βρίσκονται στον ίδιο τον πηγαίο κώδικα του προγράμματος.

# Γενικές παραδοχές

Οι παραδοχές που έγιναν κατά την ανάπτυξη του παραδοτέου κώδικα είναι οι ακόλουθες:

* Δεν υποστηρίζεται η εκτέλεση εντολών με wildcard χαρακτήρα (πχ. ‘\*’)
* Δεν υποστηρίζεται η εκτέλεση διασωληνούμενων εντολών, εκ των οποίων η δεύτερη τρέχει στο παρασκήνιο.

# Βασικές κλάσεις

Οι βασικές κλάσεις που χρησιμοποιήθηκαν είναι οι εξής:

## Scheduler

Πρόκειται για την βασική κλάση του δρομολογήτη. Κρατά εσωτερικά τα pids των διεργασιών που του έχουν υποβληθεί, ενώ εκκινείται ανά 1 sec, πραγματοποιώντας μία Round Robin δρομολόγηση σ’αυτά.

## Command

Πρόκειται για την βασική κλάση που περιγράφει μία «εντολή», δηλ. μία είσοδο του χρήστη. Περιέχει το όνομα του εκτελέσιμου προγράμματος, τις παραμέτρους αυτού, πιθανούς τελεστές, κλπ.

## CommandPrompt

Πρόκειται για την κλάση που περιγράφει το interface του shell με τον χρήστη. Δοθείσας μίας συμβολοσειράς εισόδου, δημιουργεί instances από την κλάση Command, αναγνωρίζοντας επιμέρους σημεία της (πχ. ανακατεύθυνση, pipelining, κλπ).

## Utils

# Προβλήματα

Κατά την διάρκεια της υλοποίησης, αντιμετωπίσαμε τα παρακάτω προβλήματα:

* Υλοποίηση μίας αποδοτικής συνάρτησης αποκοπής συμβολοσειρών και διαχωρισμού των λέξεων με βάση έναν χαρακτήρα (tokenize).  
    
  Αν και υπάρχουν ήδη σχετικές υλοποίησεις, καμία δεν κάλυψε τις ανάγκες της παρούσας εργασίας, γι’αυτό και οδηγηθήκαμε στην ανάπτυξη μίας εκ νέου.
* Επικοινωνία του δρομολογητή των εντολών στο παρασκήνιου με το κύριο πρόγραμμα, προκειμένου να χειρίζεται τις διεργασίες αυτές (αποστολή σημάτων SIGCONT, SIGSTOP, με αλγόριθμο Round Robin).   
    
  Αρχικά, η παρακάτω λειτουργία υλοποιήθηκε με την χρήση αρχείων στα οποία κρατώνταν τα pids των διεργασιών στο παρασκήνιο. Ωστόσο, η υλοποίηση αυτή σύντομα οδήγησε σε προβλήματα. Αρχικά, ο δρομολογητής δεν μπορούσε να στείλει σήμα SIGSTOP στις διεργασίες αυτές, καθώς δεν ήταν child processes του. Έπειτα, δημιουργόταν θέματα συναγωνισμού (race conditions) για την ανάγνωση των pids, και την δρομολόγηση των διεργασιών πρωτού ληφθεί δεύτερο σήμα στον δρομολογητή.  
    
  Το συγκεκριμένο πρόβλημα, επιλύθηκε με την εισαγωγή διπλής διασωλήνωσης μεταξύ της κύριας διεργασίας και της διεργασίας του δρομολογητή. Συγκεκριμένα, το ένα ζεύγος περιγραφέων αρχείων αφορά στην ανταλλαγή αντικειμένων τύπου «Command», ενώ το δεύτερο στην παρουσία μίας μεταβλητής που δηλώνει το πλήθος των εντολών που υπάρχουν στον περιγραφέα αρχείων. Κατ’αυτό τον τρόπο, ο δρομολογητής «διαβάζει» το πλήθος των εντολών που πρόκειται να συναντήσει στον περιγραφέα, και αν αυτό είναι θετικό, τότε και μόνο τότε προχωρά στην ανάγνωση αυτών.
* Ο τερματισμός του προγράμματος θα πρέπει να συνεπάγεται τερματισμό όλων των διεργασιών και πόρων που έχουν δεσμευτεί από το κέλυφος. Εφ’όσον ο δρομολογητής βρίσκεται σε διαφορετική διεργασία από την γραμμή εντολών, βασίσαμε την λειτουργικότητα αυτή στην χρήση ενός signal handler (σήμα τύπου SIGINT) από τον δρομολογητή.

Όταν ο τελευταίος λάβει το σήμα αυτό, τότε σκοτώνει (σήμα SIGKILL) όλες τις διεργασίες στο παρασκήνιο, απελευθερώνει τυχόν πόρους, και τερματίζει.

# Μελλοντικές επεκτάσεις

Στις μελλοντικές επεκτάσεις του προγράμματος, θα μπορούσαν να υλοποιηθούν τα ακόλουθα:

* υποστήριξη διασωλήνωσης στο παρασκήνιο με χρήση command grouping
* πολλαπλή διασωλήνωση0
* χρήση τελεστή \* (wildcard)
* αυτόματη συμπλήρωση ονομάτων εντολών και αρχείων.

# Σύνοψη

Το κέλυφος που υλοποιήθηκε ζητά εντολές από τον χρήστη έως ότου πληκτρολογηθεί exit. Κατόπιν, γίνεται parsing και τεμαχισμός της εντολής με την βοήθεια της συνάρτησης tokenize. Αφού αναγνωρισθεί η εντολή εκτελείται το ανάλογο τμήμα κώδικα. Αναφορικά, το κέλυφος υποστηρίζει τις ακόλουθες ενέργειες:

1. Αλλαγή του τρέχοντος καταλόγου σε εναλλακτικό δοθέντα κατάλογο – εντολή cd
2. Ανακατεύθυνση της εισόδου ή εξόδου μιας εντολής από ή προς ένα αρχείο – εντολές «<», «>», «>>» με ανάλογα ορίσματα
3. Διασωλήνωση μεταξύ δύο διεργασιών – τελεστής «|» μεταξύ δύο διεργασιών
4. Εκτέλεση διεργασιών στο υπόβαθρο – τελεστής «&» στο τέλος της εντολής
5. Οι διεργασίες που βρίσκονται στο υπόβαθρο δρομολογούνται σύμφωνα με τον αλγόριθμο RR (Round Robin) - χρονοπρογραμματισμός εξυπηρέτησης εκ περιτροπής