RWSH - Really Weird Shell

Un shell alternativ pentru sisteme UNIX

Tudor-Ioan Roman

Contents

Introducere - shell-ul

Shell-ul este interfața textuală a unui sistem de operare. Prin acesta, utilizatorul poate să execute programe sub formă de *comenzi*, sau să execute *script-uri*, folosind shell-ul ca interpretorul unui limbaj special de programare. Comenzile date shell-ului implică cooperarea dintre programele sistemului pentru a ajunge la un rezultat. Astfel, utilizatorul poate să prelucreze date și să administreze sistemul cu o eficiență foarte mare. Exemple de astfel de shell-uri sunt GNU `bash` (Bourne Again Shell), `csh` (C Shell), `ksh` (Korn Shell), `zsh` (Z Shell), `fish` (Friendly Interactive Shell) etc.

Prelucrarea textului

Pe platformele descendente din UNIX, precum Linux și MacOS, programele care operează în modul text se bazează pe schimbul și prelucrarea informației de tip *text simplu*, fără alte formate binare (precum pe Windows).

Exemplu: afișarea tuturor fișierelor dintr-un folder care conțin `infoeducatie` în nume, în ordine inversă:

```
ls | grep infoeducatie | sort -r
```

Această linie de comandă conține trei comenzi: `ls`, `grep infoeducatie` și `sort -r`. Cele trei comenzi sunt legate între ele prin operatorul `|` (pipe). Operatorul pipe capturează rezultatul comenzii din stânga (ce se afișează pe ecran) și, în loc să îl afișeze, îl da ca date de intrare programului din dreapta, ca și cum ar fi datele date de la tastatură.

Comanda `ls` afișează fișierele din directorul curent. `grep infoeducatie` afișează șirurile de caractere de la citire care conțin subșirul "infoeducatie", iar `sort -r` ordonează în ordine lexicografică inversă. Când comanda `ls` "afișează" fișierele, textul este dat ca intrare comenzii `grep infoeducatie`, iar aceasta la rândul ei furnizează comenzii `sort -r` ca date de intrare fișierele care in denumirea lor conțin subșirul "infoeducatie". La final, rezultatul comenzii `sort -r` este afișat pe ecran.

Acest mod de funcționare al shell-ului (*piping*) se bazează pe faptul ca majoritatea programelor operează pe text, furnizând, filtrând și prelucrând text. `ls` furnizează text, `grep` filtrează, iar `sort` prelucrează (ordonează). Programele care operează pe text includ și uneltele de administrare a sistemului. Prin

metoda *piping*-ului se pot realiza programe (*script-uri*) eficiente.

RWSH - prin ce diferă

RWSH include propriile facilități de prelucrare a textului, care operează într-un mod inedit, diferit de oricare alt shell, facilități inspirate de limbajul de prelucrare al textului folosit de `sam`, editorul de texte din sistemul de operare experimental *Plan 9*, dezvoltat în anii '80 de Laboratoarele Bell.

În mod tradițional, marea majoritate a programelor care operează pe text prelucrează datele linie cu linie. În unele cazuri, aceasta abordare poate fi ineficientă și pentru procesor, dar și pentru programator.

Expresiile regulate structurale (structural regular expressions)

RWSH folosește un sub-limbaj prin care poate fi exprimată structura textului pe care dorim se operăm. Un alt mecanism foarte important este cel al *expresiilor regulate* (regular expressions, pe scurt *regex*). Acestea sunt șiruri de caractere, exprimate într-un limbaj special, care definesc un *șablon de căutare*. Aceste expresii regulate, extinse cu facilități de descriere a structurii, dau naștere *expresiilor regulate structurale* (structural regular expressions).

Acest sub-limbaj include operații de prelucrare a textului, care pot fi combinate cu programele convenționale.

Exemplu: înlocuirea numelui "Tudor" cu "Ioan" într-un document.

```
cat document.txt |> ,x/Tudor/ c/Ioan/ |> ,p
```

Modul de funcționare

O *comandă pizza* este formată dintr-o *adresă* și o *operație*. Adresa este o expresie regulată structurală, iar operația este identificată printr-o literă și poate avea parametri. Aceste comenzi sunt înlănțuite prin operatorul `|>`, numit *operatorul pizza* (pentru că seamănă cu o felie de pizza). Adresa poate fi omisă, fiind folosită adresa ultimei comenzi, numită *dot*. Intern, adresa este o pereche de numere: poziția de început, și poziția de sfârșit, în caractere de la începutul fișierului. Dot este setată atunci când se specifică în mod explicit

adresa unei comenzi, și la finalul execuției comenzii. De exemplu, comanda `c`, care înlocuiește textul situat la adresa *dot* cu textul dat ca parametru, setează la final *dot* ca adresa la care se află textul cel nou.

În continuare, voi ilustra exemplul de mai sus:

`cat document.txt` invocă programul `cat`, care afișează pe ecran conținutul fișierului `document.txt`.

Comanda, fiind urmată de operatorul pizza (`|>`), conținutul fișierului, în loc să fie afișat pe ecran, va fi pasat comenzilor pizza care urmează.