ARHITECTURA SISTEMELOR DESCHISE

Dragoş Oprişan

Academia de Studii Economice, Bucureşti

Rezumat: Arhitectura sistemelor deschise încorporează foarte multe tehnologii diferite, înlănţuite, cum ar fi: un limbaj de comandă (sau shell-ul utilizatorului); aplicaţii incluzând utilitare şi instrumente de dezvoltare; servicii pentru aplicaţii (managementul fişierelor directoarelor, fire de tipărire etc); servicii de sistem (operaţii ale lui kernel, comunicaţii între procese, planificarea execuţiei proceselor etc); componenta hardware (UCP, Cache, RAM, memoria auxiliară, dispozitive periferice, modemuri, linii de comunicaţie etc).

Cuvinte cheie: sistem deschis, limbaj de comandă, servicii pentru aplicaţii, servicii de sistem, hardware.

1. Introducere

Structura unui sistem deschis încorporează foarte multe tehnologii deferite şi înlănţuie tehnici dintr-un număr diferit de arii din domeniul calculatoarelor. În mod evident, există foarte multe moduri în care pot fi utilizate aceste tehnici şi tehnologii.

Foarte multe domenii ale industriei cunosc schimbări rapide, care afectează arhitectura sistemelor deschise.

2. Arhitectura logică pentru sisteme deschise

Un sistem de calcul poate fi privit, în mod abstract, ca fiind format dintr-un număr de niveluri:

- *un limbaj de comandă* (sau shell-ul utilizatorului);
- *aplicaţii* (incluzând utilitare şi instrumente de dezvoltare);
- *servicii pentru aplicaţii* (gestiunea fişierelor, directoarelor, fire de tipărire etc.);
- *servicii de sistem* (operaţii ale lui Kernel, comunicaţii între procese, planificarea execuţiei proceselor etc.);
- *componenta hardware* (UCP, cache, RAM, memorie auxiliară, dispozitive periferice, modemuri, linii de comunicaţii etc.).

O vedere similară poate fi adoptată pentru sistemele deschise. În figura 1, se prezintă o imagine logică asupra unui sistem deschis:

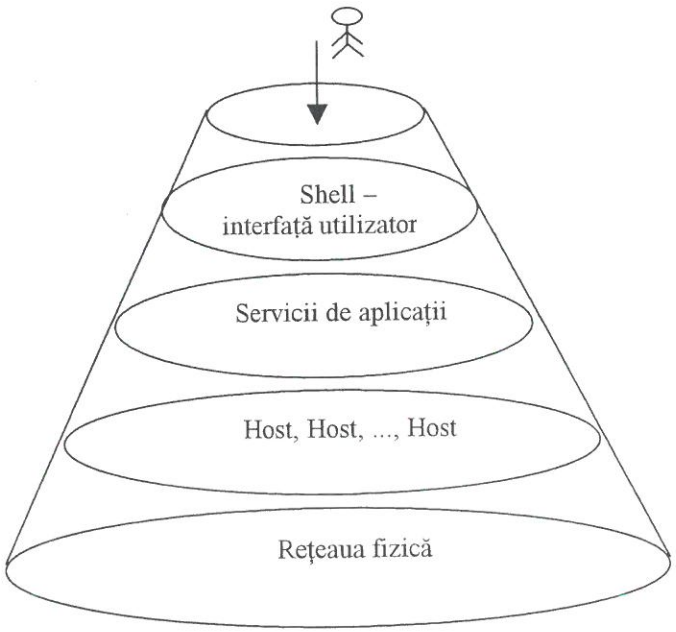

Figura 1. Arhitectura logică a unui sistem deschis

- *Shell sau interfață utilizator.* În cazul unui sistem deschis, o astfel de interfață poate fi dependentă de utilizatorul respectiv şi poate diferi de la un sistem la altul. De exemplu, administratorul de sistem are o interfață diferită de interfața unui analist financiar, cu toate că amândoi pot utiliza acelaşi sistem deschis. De asemenea, utilizatorul unui procesor care lucrează pe cuvânt, poate să folosească un microcalculator, şi un programator poate să utilizeze un mainframe. Interfețele diferă foarte mult datorită dependenței de aplicație şi sistem.
- *Nivelul de interfață*, format dintr-o *colecție de aplicații* puse la dispoziția utilizatorului. În această activitate, o dezvoltare poate fi, de asemenea, considerată a fi efectuată de un utilizator care foloseşte instrumente specializate, pentru dezvoltare adică editoare, compilatoare, depanatoare etc. Aplicațiile, la rândul lor, fac uz de servicii de aplicații (utilitare).
- *Servicii de aplicații*: oferă mecanisme şi mijloace pentru a asigura şi a ajuta execuția aplicațiilor din cadrul sistemului deschis, pe toate sistemele de calcul, care constituie componenta hardware a sistemului deschis. În particular, sunt foarte importante următoarele clase de servicii: managementul bazelor de date, serviciile de comunicație şi directoarele.
- *Serviciile pentru managementul bazelor de date.* Aceste servicii asociate bazelor de date se ocupă cu distribuirea şi managementul datelor în multiple puncte ale sistemului deschis. Aceste servicii, de asemenea, oferă mijloacele pentru utilizator şi/sau firele de aplicații asupra bazelor de date locale şi la distanță, cum ar fi servicii pentru recepționarea şi rezolvarea tranzacțiilor solicitate de aplicații, cópii de siguranță pentru date, refacerea bazei de date etc.
- *Serviciile de comunicație.* Aceste servicii oferă mijloace pentru poşta electronică, rutarea tranzacțiilor, transferul fişierelor, execuția proceselor de la distanță, procesul de logare etc. De asemenea, ele oferă o serie de mecanisme de bază, pentru transferul mesajelor între aplicații plasate la distanță în puncte diferite ale sistemelor deschise.
- *Directoarele.* În final, directoarele oferă mijloace de identificare a entităților din cadrul unui sistem. Împreună, acest set de servicii oferă baza pentru adăugarea de servicii de aplicație, cum ar fi managementul resurselor rețelei sau analiza performanțelor sistemului deschis. Acest nivel „servicii de aplicație" este un nivel utilizat de o mulțime de utilizatori (hosts). În mod practic, fiecare serviciu de aplicație este realizat printr-un set de procese pentru fiecare host, din cadrul întregului sistem. Datorită acestui fapt, acest nivel al „serviciilor de aplicație" sau „the host layer" constituie o platformă pentru toate serviciile de aplicație. Dintr-o privire sumară, se poate observa că fiecare host constă din două componente: software şi hardware, în particular, „the host operating system". Se poate observa că acest nivel are o parte logică şi o parte rețeaua fizică, fiind considerate ca o bază de conectare a tuturor utilizatorilor (hosts). Este evident faptul că „the host layer" este foarte important deoarece el îmbină aplicațiile şi serviciile de aplicații într-o rețea.

3. Rolul sistemului de operare în cadrul sistemelor deschise

În mod tradițional, un sistem de operare (SO) este considerat ca un program de control, responsabil pentru o serie de activități: managementul resurselor, planificarea resurselor şi a execuției proceselor, controlul accesului (protecția şi securitatea datelor), controlul dispozitivelor etc. În cadrul unui SO, aceste activități nu se schimbă, sistemului de operare cerându-i-se să controleze facilitățile fiecărui host particular.

Totuşi, un SO pentru un host dintr-un sistem deschis are un set extins de servicii centrate pe serviciile de aplicație.

4. Comunicațiile

Unul dintre serviciile majore ale unui SO îl constituie controlul *comunicației între procese*. În mod tipic, o astfel de comunicație între procese este 1:1, procesul emițător transmite un mesaj la un singur proces receptor şi primeşte un răspuns. În cadrul sistemului deschis, rolul SO este extins deoarece facilitățile de procesare şi de memorare pot fi divizate între diferite „hosts" conectați în cadrul unei rețele. În acest context, comunicația între procese trebuie să fie dezvoltată şi extinsă pentru a permite comunicația de la unu la mulți (1:n). O astfel de comunicație este utilă atunci când identitatea unui receptor nu este cunoscută, ca şi când noi hosts ar fi fost adăugați în sistem.

De asemenea, se oferă mijloace mai eficiente de comunicație pentru a expedia mesaje separate, în special, când componenta hardware oferă facilități eficiente, pentru transmisie (broadcasting).

Sistemul de operare trebuie să ofere facilități pentru ca „serviciile de aplicație" să permită comunicații de tipul aplicație - la-aplicație (application-to-application). Făcând apel la modelul ISO/OSI, SO host este

responsabil pentru funcţiile de la nivelul nivelurile: transport, sesiune şi prezentare. El trebuie să ofere aceste servicii pentru aplicaţii şi serviciile de aplicaţie.

În plus, serviciile de reţea şi baza de date pot necesita acces de felul facilităţilor de ceas: timpul din zi, proces suspendat/resumat, toate care trebuie furnizate de „host system".

Mai mult, aplicaţiile şi serviciile de aplicaţii vor fi realizate ca un set de programe distribuite, iar limbajele pentru implementarea acestora vor trebui să permită comunicarea şi sincronizarea activităţilor. Aceste primitive trebuie să fie furnizate de „host system" şi pot fi suportate prin facilităţi adiţionale pentru managementul firelor de aşteptare, formate din mesaje, semnalarea unui proces particular prin sosirea unui mesaj sau eroare de transmisie.

5. Managementul fişierelor

Managementul fişierelor constituie o parte care cade în sarcina SO. În sistemele deschise, sistemul de fişiere este numai o parte rezidentă a unui singur host şi, deci, nu sub controlul unui singur SO.

Acest lucru conduce la numeroase probleme incluzând: *accesul la date, controlul concurenţei* şi *controlul blocajelor*.

În literatură, există diferite soluţii pentru explorarea acestor probleme:

- *accesul la date* poate fi realizat în funcţie de diversele „unităţi" utilizate: un întreg fişier, un subset secvenţial al fişierului (Cambridge File Server), o pagină (LOCUS), un subset de pagini (XEROX Distribute File System);
- *controlul concurenţei* oferă un mijloc de accesare partajată a fişierelor. O tehnică frecventă pentru controlul concurenţei este strategia „un singur scriitor şi cititori multipli, (1-n)". O tranzacţie nu poate citi un fişier, dacă şi numai dacă fişierul nu a fost scris, şi o tranzacţie nu poate scrie un fişier, dacă şi numai dacă fişierul nu este citit sau scris. Unitatea pentru controlul concurenţei, adică entitatea cu care se operează, poate fi întreg fişierul sau o pagină;
- *controlul blocărilor (deadlock control)* poate fi realizat într-o varietate de feluri. De exemplu: FELIX file server impune ca o tranzacţie să declare toate fişierele înainte de startare şi, deci, detectează toate blocările posibile, înainte ca o tranzacţie să înceapă în sistemul XDFS. O tranzacţie solicită o pagină sau K pagini ($K \geq 2$) şi apoi aşteaptă pentru un segment de timp, înainte ca timpul afectat să se fi consumat, după care ea trebuie să obţină accesul la paginile solicitate.

Altă problemă care apare la managementul fişierelor implică alocarea fişierelor (adică, pe care host fişierul este memorat) şi acces (adică, poate un utilizator schimba host-ul şi încă să acceseze toate fişierele lui originale).

Activitatea în acest domeniu s-a concentrat, în principal, pe „hosts" având sisteme de operare omogene, adică acelaşi SO pentru fiecare sistem.

În particular, o mare problemă pentru această activitate a fost realizată utilizând UNIX, adică Sun Network File System, LOCUS.

O colecţie eterogenă de *hosts* prezintă, uneori, o serie de probleme mari.

6. Domeniul de nume

Hosts în sistemele deschise sunt influenţate de problemele denumirii, relativ la problemele managementului fişierelor. În cazul fişierelor, de exemplu, un set de hosts omogeni implică faptul că fiecare host individual are un nume spaţiu fişier similar. Acest lucru poate fi extins într-un mod consistent, care oferă într-o reţea mare de nume, spaţiu fişier.

Alte probleme privind numele apar în următoarele situaţii: numele individual al locurilor, numele aplicaţiilor, numele proceselor. Din nou, un mediu omogen simplifică problemele. Într-un mediu eterogen, hosts li se poate cere să mapeze numele pentru diferiţi hosts pentru propriul nume spaţiu şi invers.

În concluzie, directoarele pot fi utilizate să realizeze o simplificare a acestui set de probleme particulare.

7. Suport pentru baze de date (BD)

Controlul concurenței oferă un mecanism necesar pentru acces partajat la date. Acest lucru este, de asemenea, foarte necesar pentru managementul BD.

În plus, sistemele de operare host trebuie, de asemenea, să ofere realizarea tranzacțiilor, pentru acces la BD, și actualizarea datelor în BD. Aceasta, în final, necesită faptul ca un singur protocol să fie utilizat la intrarea în sistem a „hosts".

8. Sisteme de operare pentru sisteme deschise

Este evident faptul că o colecție de sisteme de calcul eterogene are un număr de probleme dificile.

Întâmplător, este de dorit să fie o varietate de tipuri de sisteme într-un sistem deschis.

9. Set limitat de sisteme de operare

Dacă anumite probleme pot fi mai ușor de manipulat cu un singur sistem de operare pentru toți „hosts", atunci, probabil că, lucrând numai cu un număr mic de sisteme de operare, nu pot introduce așa multe activități complexe.

De exemplu, arhitectura de rețea Intel's Open Net permite sisteme hosts care pot rula pe RMX sau XENIX.

10. Sisteme de operare portabile

Un lucru evident este să se realizeze cópii cu diferite sisteme host, să se selecteze un singur SO și acesta să fie portabil pe fiecare host.

O preocupare serioasă a fost aceea de a se dezvolta sisteme de operare portabile.

Parseus a constituit un experiment interesant în dezvoltarea de sisteme de operare portabile pentru un sistem de calcul distribuit, implicând diferite hosts (incluzând DecSystem-10, VAX și IBM 4331). Obiectivul a fost de a construi un sistem de operare de interes general, care poate fi portabil pe un domeniu de mașini. Sistemul de operare a fost implementat într-o versiune PASCAL, facilitând sisteme de programare, manipulări de excepții și compilare separată.

În principiu, pentru a putea realiza acest lucru, se poate considera că un sistem de operare existent sau proiectarea uni sistem de operare, care oferă multe facilități dorite, este simplu, bine structurat și probabil scris într-un limbaj de nivel înalt.

Cu astfel de trăsături, este ușor de portat un sistem de operare pe un nou hardware. Un sistem de operare existent pe piață, care poate fi un candidat puternic, este UNIX, el rulând pe toate tipurile de calculatoare (de la microcalculatoare la mainframe), fiind portabil pe o varietate mare de hardware, fiind scris în limbajul C și oferind un număr de facilități de bază.

O preocupare o constituie realizarea unui SO, în general, urmărind o structurare pentru dezvoltarea sistemelor de operare actuale. O soluție adoptată în comun este structurarea sistemului de operare, văzut ca nu număr de mașini virtuale (figura 2).

Pentru fiecare din mașinile virtuale, mașina K este implementată pe mașina K-1. Mașina 0 va corespunde la hardware-ul actual și mașina n - ca o vedere a utilizatorului drept sistem.

În cazul când interfețele dintre mașini sunt riguros specificate și bine definite, este necesară implementarea mașinii 1 pe mașina 0 cu scopul de a porta sistemul de operare. Evident, se presupune că celelalte mașini sunt implementate într-un limbaj care este portabil și că programul compilator există pe mașina 0.

Această soluție pentru dezvoltarea sistemelor de operare portabile este definită în forme variate de către un număr mare de cercetători. Este, de fapt, un mod similar cu felul în care SO UNIX este implementat: facilitățile oferite de componenta Kernel a lui UNIX corespunde „mașinii 1" din figura 2, iar restul mașinilor din figura 2, sunt definite ca biblioteci C, de funcții și programe.

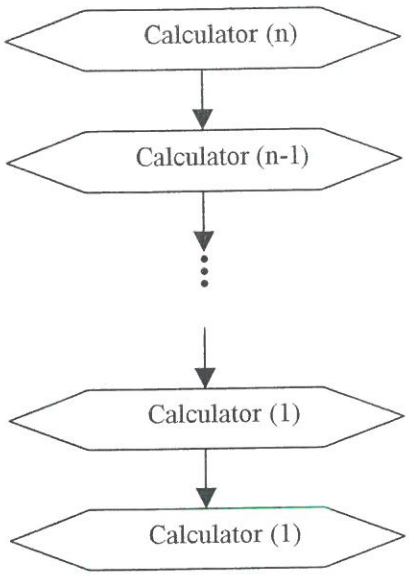

Figura 2. Prezentarea unui calculator virtual

11. Sisteme de operare virtuale

O altă alternativă de a dezvolta un SO portabil este aceea de a defini un sistem de operare virtual. În astfel de situații, se specifică un set de: obiective ale sistemului, proprietăți și funcții aparținând unui sistem de operare ipotetic.

Aceste obiective și funcții sunt la îndemâna tuturor aplicațiilor și, mai mult, sunt numai funcțiile și obiectivele „sistemului de operare", pe care aplicația le poate utiliza. La rândul său, sistemul de operare virtual este implementat la nivelul superior al sistemelor de operare host. În contrast cu un SO portabil, se poate alege o implementare a sistemului de operare virtual, pentru care mașina „0" din figura 2 este un SO host existent. O astfel de abordare poate să nu solicite multe mașini virtuale.

De asemenea, el poate fi mai puțin complex deoarece:

- el poate solicita numai facilitățile de nivel înalt ale SO (cum ar fi crearea fișierului, alocarea memoriei);
- el necesită suportarea aplicațiilor și serviciilor de aplicație dacă se presupune că arhitectura unui sistem deschis este cea dată în figura 1. Mai mult, actuala portabilitate a acestei mașini virtuale poate fi simplificată dacă fiecare din mașinile host suportă un limbaj standard, cu facilități pentru utilizarea unei astfel de mașini virtuale (MODULA II, ADA).

12. Impactul tehnologic al arhitecturii sistemelor deschise

În lumina evoluției rapide a tehnologiei, este evident faptul că efectele sunt numeroase schimbări în arhitectura sistemelor deschise.

13. Very Large Scale Integration (VLSI)

Utilizarea pe scară largă a VLSI a avut drept rezultat creșterea puterii de calcul a microcalculatoarelor, o componentă hardware mult mai performantă și specializată (atât pentru aplicații generale, cât și dedicate), precum și realizarea unor arhitecturi mult mai sofisticate.

De asemenea, aceste progrese au condus la schimbări evidente pentru hosts, stații de lucru și alte dispozitive inteligente, care au avut ca rezultate dezvoltarea sistemelor deschise:

- *în primul rând*, realizările hardware, unde au apărut noi procesoare, software performant cum ar fi compilatoare sau translatoare de comunicație, într-o formă de silicon, precum cererea de realizare a anumitor standarde să fie adoptate. Faptul că arhitectura sistemelor deschise necesită utilizarea standardelor acolo unde este posibil, va servi la stabilizarea anumitor domenii. Această organizare pe niveluri a sistemelor deschise este echivalentă cu faptul că diverse servicii între niveluri, adică anumite SO sau servicii de comunicație, pot fi realizate la nivel hardware mai mult decât la nivel software.

- *În al doilea rând*, progresele în tehnologia hardware vor servi la realizarea unor probleme la nivelul performanţelor asociate cu diverse standarde. În comunicaţii, de exemplu, standardele au fost definite şi acceptate. Tehnologia va permite ca aceste standarde să fie implementate în cadrul componentei hardware care, în prezent, este mult mai performantă. În anumite domenii, cum ar fi comunicaţiile, aceste facilităţi constituie, în prezent, o realitate.

14. Arhitecturi evoluate pentru calculatoare

Dorinţa de a realiza arhitecturi de calcul evoluate cum ar fi: calculatoare cu prelucrare în flux (dataflow), calculatoare paralele, calculatoare pentru prelucrarea cunoştinţelor, va ridica, din nou, o serie de probleme pentru realizarea sistemelor deschise.

Pe de altă parte, astfel de calculatoare oferă posibilitatea realizării unor posibile servere de aplicaţii puternice şi/sau specializate.

Sistemele deschise pot fi realizate ca resurse, ca dispozitive de calcul, disponibile pentru un număr de utilizatori, sau să fie încorporate într-un SO, astfel încât utilizatorul să nu fie interesat de orice schimbare de la nivelul hardware.

14.1. Dispozitive evoluate

De asemenea, progresele înregistrate la nivelul tehnologiei deţin un potenţial evident privind o diversitate de dispozitive specializate şi inteligente, dispozitive periferice performante, precum: „digitizers" şi discuri optice.

Un hardware şi un software de comunicaţie standardizat realizează o serie de dispozitive dedicate pentru utilizatorii sistemelor deschise. Încorporarea unor astfel de dispozitive în sistemele deschise poate fi realizată într-o astfel de manieră încât utilizatorul să nu fie îngrijorat de prezenţa lor.

De exemplu, utilizarea unui disc optic ca memorie fizică într-o bază de date implică înlocuirea discului magnetic, dar încă prezenţa lui este transparentă pentru utilizator şi pentru serviciile bazei de date însăşi.

Bibliografie

1. **GOLDAM, J.E.**: Local Area Networks A CLIENT/ Server Apprach.
2. **JAY, M.**: Multiprocessor Servers. Network Computing, 1994.
3. **PADRAIC, B.**: PC Magazine, 1995.
4. **CHIANG, AL.**: Fast Ethernet Comunications System Design, Aug 1995.
5. **O'LEARY, T.J., O'LEARY, L.J.**: Computing Essentials, McGraw - Hill, 1996 - 1997.
6. **O'LEARY, T.J., O'LEARY, L.J:** Computing Essentials, Multimedia Edition 1997-1998.
7. **CHARNEY, H.**: Lan Architecture and Design, 1994.
8. **SLONIM, J., BAUER, M.A.**: Building An Oper System, Van Nostrand Reinhold, 1985.
9. **GRAY, J. N.**: A discusion of Distribute System, IBM Research Raport, 1989.