Κατανεμημένα Συστήματα
Επικοινωνία
Client/Server

Χάρης Μανιφάβας
Τμήμα Εφ. Πληροφορικής & Πολυμέσων
ΤΕΙ Κρήτης

Μοντέλο Πελάτη-Εξυπηρετητή

- Βασική ιδέα: να δομηθεί το λειτουργικό σύστημα ως συνεργαζόμενες διεργασίες
  - Τους εξυπηρετητές (ή διακομιστές) που προσφέρουν τις υπηρεσίες τους
    - π.χ. υπηρεσία συστήματος αρχείων, υπηρεσία βάσης δεδομένων
  - Τους πελάτες που είναι οι χρήστες των υπηρεσιών

- Ένα σύστημα client-server είναι ένα σύστημα στο οποίο το δίκτυο ενώνει διάφορους υπολογιστικούς πόρους
  - Έτσι οι clients (ή αλλιώς front end) μπορούν να ζητούν υπηρεσίες από έναν server (ή αλλιώς back end), ο οποίος προσφέρει πληροφορίες ή επιπρόσθετη υπολογιστική ισχύ

Μοντέλο Πελάτη-Εξυπηρετητή

- Το μοντέλο βασίζεται, συνήθως, σε ένα απλό πρωτόκολλο αίτησης/απάντησης (request/reply)
  - Ο πελάτης στέλνει μήνυμα αίτησης ζητώντας από τον εξυπηρετητή κάποια υπηρεσία
  - Ο εξυπηρετητής εκτελεί τη διαδικασία και επιστρέφει τα δεδομένα που ζητήθηκαν ή ένα μήνυμα λάθους
  - Με άλλα λόγια, στο client-server μοντέλο, ο client θέτει μια αίτηση και ο server επιστρέφει μια ανταπόκριση ή κάνει μια σειρά από ενέργειες
  - Ο server μπορεί να ενεργοποιείται άμεσα για την αίτηση αυτή ή να προσθέτει την αίτηση σε μια ουρά

Μοντέλο Πελάτη-Εξυπηρετητή

- Το πρωτόκολλο αίτησης / απάντησης (request / reply protocol) από λογικής άποψης, δεν είναι προσανατολισμένο σε σύνδεση
  - Στο πρωτόκολλο αυτό, ο πελάτης στέλνει ένα μήνυμα αίτησης προς τον εξυπηρετητή, ζητώντας να του παρασχεθεί κάποια υπηρεσία
  - Ο εξυπηρετητής, που αναμένει συνεχώς αιτήσεις από τους πελάτες, παραλαμβάνει το μήνυμα αυτό, διεκπεραιώνει την αίτηση του πελάτη και του στέλνει ένα μήνυμα απάντησης
  - Το μήνυμα αυτό περιέχει τα δεδομένα που ζήτησε ο πελάτης ή την πληροφορία ότι η αίτησή του δεν ήταν δυνατό να διεκπεραιωθεί μαζί με κάποια αιτιολογία

Μοντέλο Πελάτη-Εξυπηρετητή

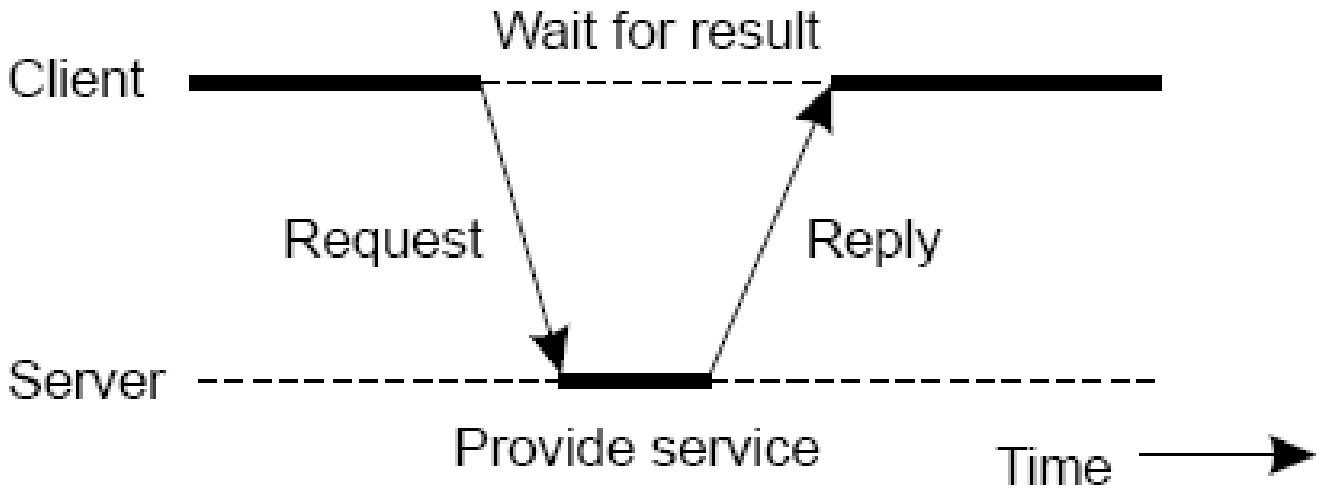
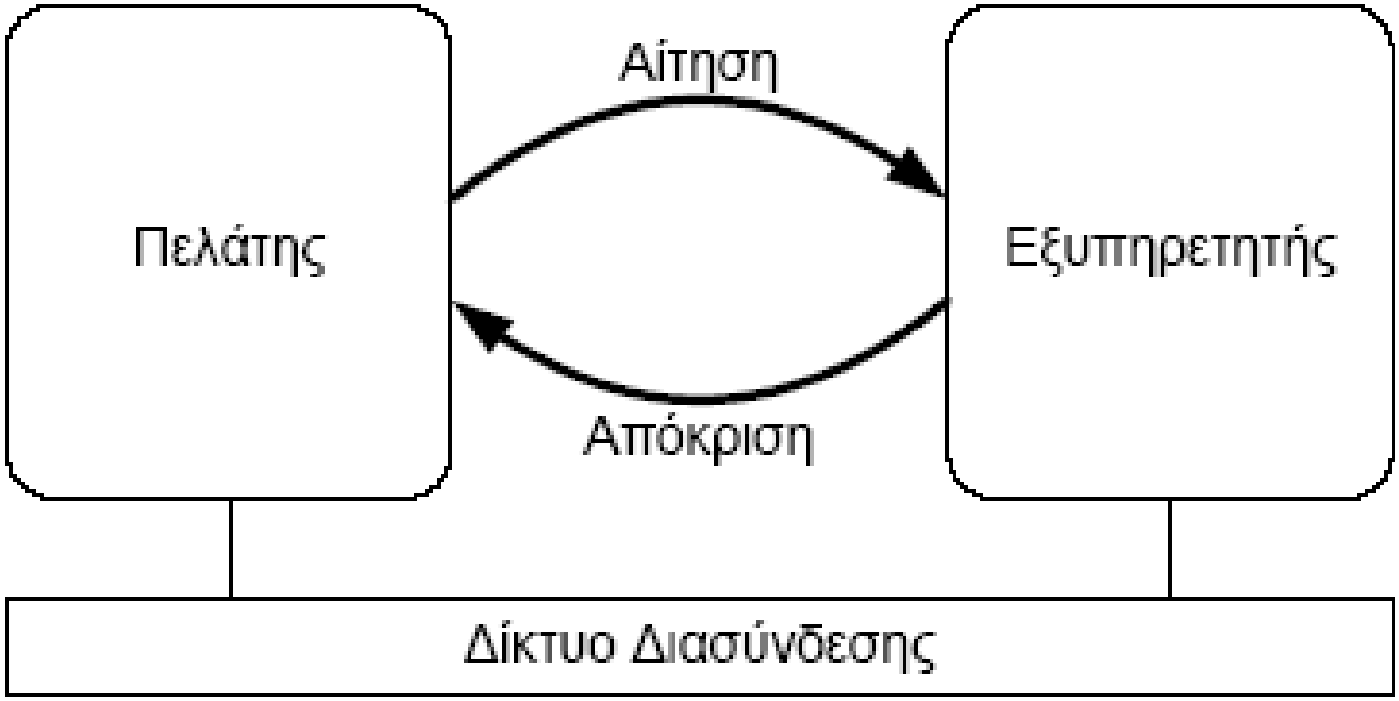


Fig. 1-25. General interaction between a client and a server.

Μοντέλο Πελάτη-Εξυπηρετητή



Σχήμα 1.5. Μοντέλο πελάτη-εξυπηρετητή.

Μοντέλο Πελάτη-Εξυπηρετητή

- Οι servers μπορούν να διαιρεθούν σε
  - Server Εφαρμογών (Application servers)
  - Server Πληροφοριών (Data servers)
  - Server Υπολογισμών (Compute servers)
  - Server Βάσεων Δεδομένων (Database servers)
  - Server Πόρων ή Επικοινωνιών (Resource or Communications servers)

  - Οι servers που συνδυάζουν τις λειτουργίες του server βάσεων δεδομένων και του server εφαρμογών είναι επίσης γνωστοί ως server συναλλαγών (transaction servers)

Μοντέλο Πελάτη-Εξυπηρετητή

- Πλεονεκτήματα
  - Ο server προσφέρει συγκεντρωμένο έλεγχο των κοινών πόρων
  - Επειδή ο client επικοινωνεί με τον server μέσω ενός καθορισμένου συστήματος διασύνδεσης, δεν χρειάζεται να γνωρίζει που ανήκει ο server ή πως ενεργεί
  - Η διαχείριση της διασύνδεσης των χρηστών και άλλες επεξεργασίες είναι αποφορτισμένα από τον «οικοδεσπότη»
    - Ο σταθμός εργασίας τρέχει την εφαρμογή και εμφανίζει τις πληροφορίες στον χρήστη
    - Μόνο όταν ο client προσπελάζει πληροφορίες, τότε εγκαθίσταται επικοινωνία με τον server
    - Ο φόρτος εργασίας μειώνεται δραματικά στον υπολογιστή-«οικοδεσπότη» όσο αυξάνεται η ισχύς κάθε σταθμού εργασίας

Πρωτόκολλα Επικοινωνίας

- Για να είναι αποτελεσματική η επικοινωνία ο αποστολέας και ο παραλήπτης πρέπει να συμφωνούν στη σημασία των πληροφοριών που ανταλλάσσονται μεταξύ τους
  - Οι πελάτες και οι εξυπηρετητές μπορεί να εκτελούνται στην ίδια ή σε διαφορετικές μηχανές, πάνω από το ίδιο ή διαφορετικά λειτουργικά συστήματα, αρκεί να υπάρχει συμφωνία μεταξύ τους στην αναπαράσταση και την ερμηνεία των μηνυμάτων που ανταλλάσσονται
  - Η συμφωνία αυτή πρέπει να λάβει χώρα σε διάφορα επίπεδα, από το κατώτερο, της μετάδοσης των δυφίων μέσω των φυσικών συνδέσεων του δικτύου, έως το υψηλότερο, της ερμηνείας των μηνυμάτων από τις επικοινωνούσες εφαρμογές
  - Συνήθως θεωρούμε ότι η επικοινωνία μέχρι και το επίπεδο δικτύου παρέχεται από το πρωτόκολλο IP, ενώ σε ανώτερα επίπεδα χρησιμοποιούνται είτε τα γνωστά πρωτόκολλα του Internet (TCP και UDP) είτε άλλα πρωτόκολλα

Πρωτόκολλα Επικοινωνίας

- Εάν το χρησιμοποιούμενο δίκτυο είναι αξιόπιστο, όπως τα ενσύρματα τοπικά δίκτυα, η επικοινωνία πελάτη-εξυπηρετητή υλοποιείται χωρίς αποκατάσταση συνδέσεων
  - Το πλεονέκτημα είναι η απλότητα της επικοινωνίας: ο πελάτης στέλνει μία αίτηση και παίρνει πίσω μία απάντηση, χωρίς να χρειάζεται να αποκατασταθεί ούτε να τερματισθεί μία σύνδεση
    - Το μήνυμα απάντησης μπορεί να χρησιμοποιηθεί ως υπονοούμενο μήνυμα *παραδοχής (acknowledgement)* της αίτησης
  - Η απλότητα του μοντέλου αυτού σημαίνει μικρότερη επιβάρυνση στο σύστημα
  - Το πρωτόκολλο αυτό ορίζει το σύνολο των έγκυρων αιτήσεων και το σύνολο των έγκυρων απαντήσεων στις αιτήσεις αυτές

Πρωτόκολλα Επικοινωνίας

- Σε ένα ΚΣ όπου οι διάφορες μηχανές είναι γεωγραφικά απομακρυσμένες, η επικοινωνία δεν είναι αξιόπιστη, με αποτέλεσμα κάποια μηνύματα να χάνονται ή να παραμορφώνονται
  - Για να αυξηθεί η αξιοπιστία του πρωτοκόλλου, ο πελάτης μπορεί να αναμεταδίδει την αίτησή του αν δε λάβει την αντίστοιχη απάντηση μέσα σε κάποιο χρονικό διάστημα
  - Το πρόβλημα είναι ότι τότε δεν μπορούμε να διακρίνουμε το κατά πόσο η έλλειψη απάντησης οφείλεται σε απώλεια της αίτησης ή της απόκρισης
  - Αν χάθηκε η αίτηση, η εκ νέου μετάδοσή της δε δημιουργεί κανένα πρόβλημα
  - Αν όμως χάθηκε η απόκριση, ο εξυπηρετητής θα έχει ήδη εκτελέσει κάποιες ενέργειες για να εξυπηρετήσει το αίτημα
    - Η αναμετάδοση της αίτησης μπορεί έτσι να οδηγήσει σε διπλή εκτέλεση των ενεργειών
    - Το κατά πόσον αυτό είναι προβληματικό εξαρτάται από την αίτηση
    - Αν η αίτηση αφορά την ανάγνωση στοιχείων, δεν έχουμε πρόβλημα, αν όμως αφορά την προσθήκη στοιχείων στο τέλος ενός αρχείου, η επανάληψη της αίτησης οδηγεί σε ανεπιθύμητη αλλαγή του αρχείου

Πρωτόκολλα Επικοινωνίας

- Για να αποφύγουμε την ενασχόληση με αυτά τα προβλήματα στο επίπεδο εφαρμογής, στα δίκτυα ευρείας περιοχής συνήθως χρησιμοποιούμε αξιόπιστα πρωτόκολλα όπως το TCP
  - Αυτό σημαίνει ότι κάθε επικοινωνία απαιτεί την εγκαθίδρυση μίας σύνδεσης ανάμεσα στον πελάτη και τον εξυπηρετητή, την αποστολή της αίτησης και την επιστροφή της απάντησης πάνω από τη σύνδεση αυτή, και τέλος την καταστροφή της σύνδεσης
  - Το πρόβλημα είναι μεγαλύτερο όταν τα μηνύματα της αίτησης και της απάντησης είναι μικρά
  - Αν και η εγκαθίδρυση και καταστροφή συνδέσεων εισάγει σημαντική καθυστέρηση στο πρωτόκολλο, η μόνη εναλλακτική λύση για μη αξιόπιστα δίκτυα είναι η αύξηση της πολυπλοκότητας του πρωτοκόλλου πελάτη-εξυπηρετητή

Μοντέλο Πελάτη-Εξυπηρετητή

Τύπος	Από	Προς	Περιγραφή
Αίτηση	Π	Ε	Ο πελάτης ζητά υπηρεσίες
Απάντηση	Ε	Π	Ο εξυπηρετητής απαντά στον πελάτη
Επιβεβαίωση	Ε/Π	Π/Ε	Το προηγούμενο πακέτο έχει φτάσει
Έλεγχος ύπαρξης	Π	Ε	Έλεγχος αν ο εξυπηρετητής έχει καταρρεύσει
Επιβεβαίωση ύπαρξης	Ε	Π	Ο εξυπηρετητής λειτουργεί κανονικά
Επανάληψη προσπάθειας	Ε	Π	Ο εξυπηρετητής δεν έχει χώρο
Άγνωστη διεύθυνση	Ε	Π	Δεν υπάρχει διεργασία που να χρησιμοποιεί αυτή τη διεύθυνση

Παράδειγμα Τοπικής Αντιγραφής Αρχείου

```
/* File copy program. Error checking and reporting is minimal. */

#include <sys/types.h>                  /* include necessary header files */
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[]);       /* ANSI prototype */

#define BUF_SIZE 4096                   /* use a buffer size of 4096 bytes */
#define OUTPUT_MODE 0700                /* protection bits for output file */

int main(int argc, char *argv[])
{
      int in_fd, out_fd, rd_count, wt_count;
      char buffer[BUF_SIZE];

      if (argc != 3) exit(1);           /* syntax error if argc is not 3 */
```

Παράδειγμα Τοπικής Αντιγραφής Αρχείου

```
/* Open the input file and create the output file */
in_fd = open(argv[1], O_RDONLY);    /* open the source file */
if (in_fd < 0) exit(2);             /* if it cannot be opened, exit */
out_fd = creat(argv[2], OUTPUT_MODE);  /* create the destination file */
if (out_fd < 0) exit(3);            /* if it cannot be created, exit */

/* Copy loop */
while (TRUE) {
    rd_count = read(in_fd, buffer, BUF_SIZE); /* read a block of data */
if (rd_count <= 0) break;           /* if end of file or error, exit loop */
    wt_count = write(out_fd, buffer, rd_count); /* write data */
    if (wt_count <= 0) exit(4);     /* wt_count <= 0 is an error */
}

/* Close the files */
close(in_fd);
close(out_fd);
if (rd_count == 0)                  /* no error on last read */
    exit(0);
else
    exit(5);                        /* error on last read */
}
```

Παράδειγμα Απομακρυσμένης Αντιγραφής

- Πελάτης – Διακομιστής αρχείων σε γλωσσα C
  - Ο πελάτης αντιγράφει ένα αρχείο χρησιμοποιώντας τον διακομιστή
  - Μοιράζονται κάποιους κοινούς ορισμούς (header.h)
    - Συμπεριλαμβάνουν αυτούς τους ορισμούς στον κώδικά τους (πριν ο compiler αρχίσει την μεταγλώτισση του προγράμματος) χρησιμοποιώντας την εντολή «#include <header.h>»
  - Σχήμα 1: header.h
  - Σχήμα 2: server
  - Σχήμα 3: client

Παράδειγμα Απομακρυσμένης Αντιγραφής

```
/* Definitions needed by clients and servers.      */
#define TRUE            1
#define MAX_PATH        255   /* maximum length of file name          */
#define BUF_SIZE        1024  /* how much data to transfer at once    */
#define FILE_SERVER     243   /* file server's network address        */

/* Definitions of the allowed operations */
#define CREATE          1     /* create a new file                    */
#define READ            2     /* read data from a file and return it  */
#define WRITE           3     /* write data to a file                 */
#define DELETE          4     /* delete an existing file              */

/* Error codes. */
#define OK              0     /* operation performed correctly        */
#define E_BAD_OPER      -1    /* unknown operation requested          */
#define E_BAD_PARAM     -2    /* error in a parameter                 */
#define E_IO            -3    /* disk error or other I/O error        */

/* Definition of the message format. */
struct message {
    long source;              /* sender's identity                    */
    long dest;                /* receiver's identity                  */
    long opcode;              /* requested operation                  */
    long count;               /* number of bytes to transfer          */
    long offset;              /* position in file to start I/O        */
    long result;              /* result of the operation              */
    char name[MAX_PATH];      /* name of file being operated on       */
    char data[BUF_SIZE];      /* data to be read or written           */
};
```

Παράδειγμα Απομακρυσμένης Αντιγραφής

```
#include <header.h>
void main(void) {
     struct message m1, m2;          /* incoming/outgoing messages */
     int r;                          /* result code                */
     while(TRUE) {                   /* server runs forever        */
          receive(FILE_SERVER, &m1); /* block waiting for a message*/
          switch(m1.opcode) {        /* dispatch on type of request */
               case CREATE:          r = do_create(&m1, &m2); break;
               case READ:            r = do_read(&m1, &m2); break;
               case WRITE:           r = do_write(&m1, &m2); break;
               case DELETE:          r = do_delete(&m1, &m2); break;
               default:              r = E_BAD_OPER;
          }
          m2.result = r;             /* return result to client    */
          send(m1.source, &m2);      /* send reply                 */
     }
}
```

(a)

Παράδειγμα Απομακρυσμένης Αντιγραφής

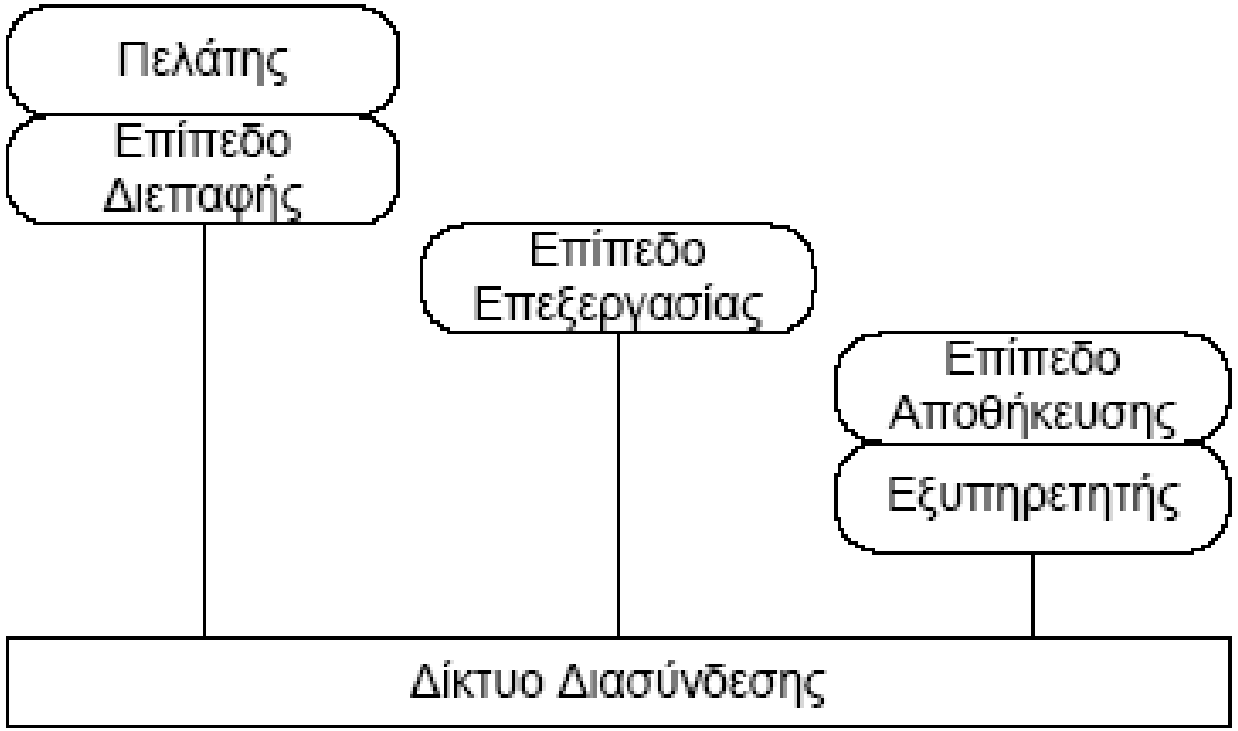
```
#include <header.h>
int copy(char *src, char *dst){          /* proc. to copy file using the server*/
    struct message ml;                   /* message buffer                     */
    long position;                       /* current file position              */
    long client = 110;                   /* client's address                   */
    initialize( );                       /* prepare for execution              */
    position = 0;
    do {
        ml.opcode = READ;                /* operation is a read                */
        ml.offset = position;            /* current position in the file       */
        ml.count  = BUF_SIZE;            /* how many bytes to read             */
        strcpy(&ml.name, src);           /* copy name of file to be read       */
        send(FILESERVER, &ml);           /* send message to the file server*/
        receive(client, &ml);            /* block waiting for the reply        */

        /* Write the data just received to the destination file.               */
        ml.opcode = WRITE;               /* operation is a write               */
        ml.offset = position;            /* current position in the file       */
        ml.count  = ml.result;           /* how many bytes to write            */
        strcpy(&ml.name, dst);           /* copy name of file to be written*/
        send(FILE_SERVER, &ml); /* send message to the file server*/
        receive(client, &ml);            /* block waiting for the reply        */
        position += ml.result;           /* ml.result is #bytes written        */
    } while( ml.result > 0 );            /* iterate until done                 */
    return(ml.result >= 0 ? OK : ml result); /* return OK or error code*/
}
```

(b)

Μοντέλο Π-Ε Τριών Επιπέδων

- Η διάκριση ανάμεσα στους πελάτες και τους εξυπηρετητές είναι πολλές φορές ασαφής
  - Για παράδειγμα, ένας εξυπηρετητής βάσεων δεδομένων μπορεί να είναι ταυτόχρονα και πελάτης ενός εξυπηρετητή αρχείων στον οποίο αποθηκεύονται οι πίνακες της βάσης δεδομένων, λειτουργώντας ως μεσάζων ανάμεσα στους πελάτες της βάσης και τους εξυπηρετητές αρχείων
  - Ο εξυπηρετητής βάσεων δεδομένων παρέχει την επεξεργασία των πρωτογενών δεδομένων έτσι ώστε να ικανοποιούνται τα αιτήματα των πελατών
  - Στην πράξη η οργάνωση πελατών και εξυπηρετητών σε τρία επίπεδα είναι η πλέον συνηθισμένη σε συστήματα διαχείρισης κατανεμημένων βάσεων δεδομένων (η οποία είναι μια κοινή εφαρμογή)
    - Το πρώτο επίπεδο παρέχει τη *διεπαφή με το χρήστη (user interface)*
    - Το δεύτερο την *επεξεργασία των δεδομένων (data processing)*
    - Το τρίτο την *αποθήκευση των δεδομένων (data storage)*

Μοντέλο Πελάτη-Εξυπηρετητή



Σχήμα 1.6. Μοντέλο πελάτη – εξυπηρετητή τριών επιπέδων.

Μοντέλο Π-Ε Τριών Επιπέδων

- Το επίπεδο διεπαφής
  - Έχει καθαρά ρόλο πελάτη και περιέχει τις εφαρμογές που επιτρέπουν στους χρήστες να αλληλεπιδρούν με την εφαρμογή και τη βάση δεδομένων
  - Η πολυπλοκότητα του επιπέδου αυτού μπορεί να ποικίλει σημαντικά
  - Στην απλούστερη περίπτωση η διεπαφή προσομοιώνει ένα τερματικό κειμένου, με το επίπεδο αυτό να περιορίζεται σε καθήκοντα όπως επεξεργασία χαρακτήρων διαγραφής και σχεδιασμό φορμών
  - Η πιο συνηθισμένη περίπτωση είναι η υποστήριξη μίας γραφικής διεπαφής με παράθυρα, μενού και πλαίσια διαλόγου
    - Η εφαρμογή διαχειρίζεται την οθόνη και τις συσκευές εισόδου του χρήστη (ποντίκι και πληκτρολόγιο) και επικοινωνεί με το επόμενο επίπεδο μόνο για να στείλει τις επερωτήσεις του χρήστη και για να παρουσιάσει στην οθόνη τα στοιχεία που επιστρέφονται από τη βάση

Μοντέλο Π-Ε Τριών Επιπέδων

- Το επίπεδο επεξεργασίας
  - Παρέχει μια γέφυρα ανάμεσα στη διεπαφή με το χρήστη και την αποθήκη δεδομένων, η οποία μπορεί να παρέχεται από εξυπηρετητές αρχείων ή εξυπηρετητές βάσεων δεδομένων
  - Τα καθήκοντα του επιπέδου επεξεργασίας εξαρτώνται από την υλοποιούμενη εφαρμογή, αλλά ο στόχος του είναι να αντλεί στοιχεία από πληροφοριακά συστήματα γενικής χρήσεως και να τα επεξεργάζεται έτσι ώστε να ικανοποιεί τις απαιτήσεις της εφαρμογής
  - Η χρησιμότητα της διάκρισης των ρόλων αποθήκευσης και επεξεργασίας έγκειται στο ότι τα δεδομένα μπορούν να χρησιμοποιηθούν με διαφορετικούς τρόπους
    - Για παράδειγμα, οι μετα-μηχανές αναζήτησης παρέχουν ένα νέο επίπεδο επεξεργασίας το οποίο μετατρέπει τις ερωτήσεις του χρήστη σε επερωτήσεις προς πολλαπλές μηχανές αναζήτησης, αξιολογεί τα αποτελέσματα και επιστρέφει μία πιο σύνθετη ιστοσελίδα στο χρήστη

Μοντέλο Π-Ε Τριών Επιπέδων

- Το επίπεδο επεξεργασίας
  - Παράδειγμα
    - Μία μηχανή αναζήτησης περιέχει μια τεράστια ΒΔ η οποία περιέχει δεικτοδοτημένες σελίδες που έχουν ήδη αναλυθεί
    - Όταν ο χρήστης εκτελεί μία αναζήτηση, ουσιαστικά στέλνει μέσω του *browser,* ο οποίος παρέχει το επίπεδο διεπαφής, μία σειρά από λέξεις κλειδιά στο επίπεδο επεξεργασίας
    - Το επίπεδο επεξεργασίας μετασχηματίζει τις λέξεις αυτές σε επερωτήσεις προς τη ΒΔ, συγκεντρώνει τα αποτελέσματα που επιστρέφονται, τα ταξινομεί με βάση κάποια κριτήρια και τα επιστρέφει στο *browser* του χρήστη με τη μορφή ιστοσελίδων

Μοντέλο Π-Ε Τριών Επιπέδων

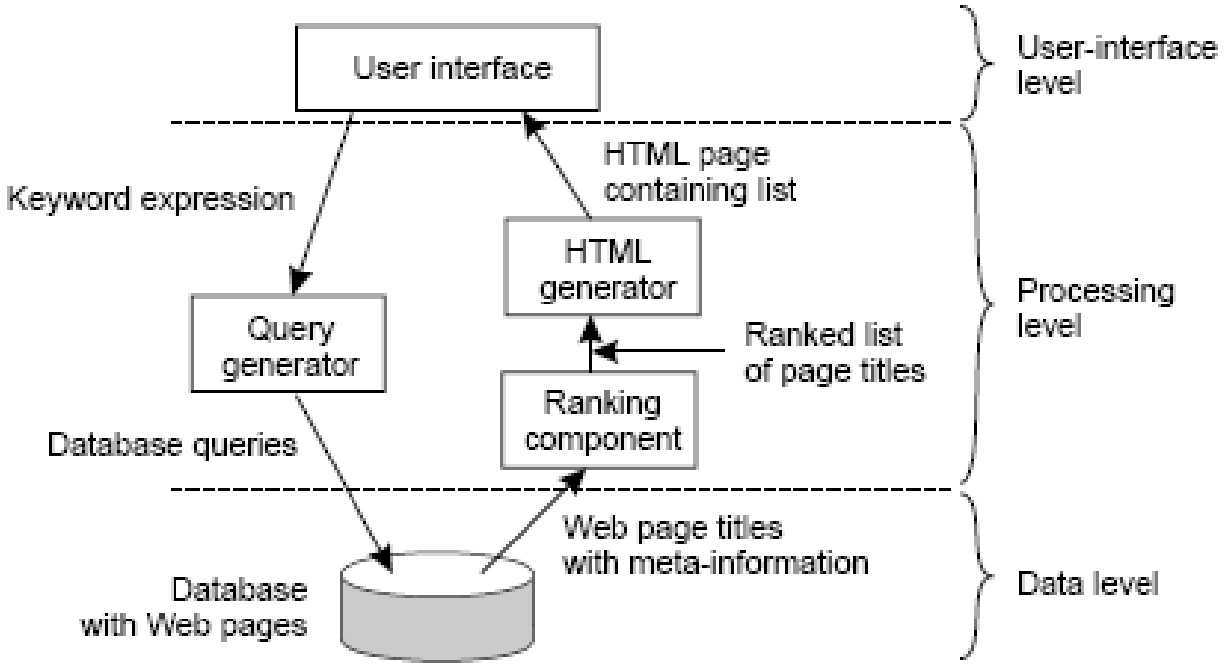


Fig. 1-28. The general organization of an Internet search engine into three different layers.

Μοντέλο Π-Ε Τριών Επιπέδων

- Το επίπεδο αποθήκευσης
  - Έχει καθαρά ρόλο εξυπηρετητή, αφού περιέχει τις εφαρμογές που διαχειρίζονται τα δεδομένα στα οποία τελικά αποκτά πρόσβαση ο χρήστης
  - Το επίπεδο αυτό είναι ανεξάρτητο από το επίπεδο επεξεργασίας, με την έννοια ότι τα δεδομένα που αποθηκεύονται είναι διαθέσιμα ακόμη και όταν η εφαρμογή επεξεργασίας δε χρησιμοποιείται
  - Έτσι ότι το ίδιο επίπεδο αποθήκευσης μπορεί να χρησιμοποιηθεί με πολλά διαφορετικά επίπεδα επεξεργασίας, πράγμα που συμβαίνει πολύ συχνά στην πράξη
  - Το επίπεδο αποθήκευσης μπορεί να είναι είτε ένα σύστημα αρχείων είτε μία ολοκληρωμένη βάση δεδομένων

Μοντέλο Π-Ε Τριών Επιπέδων

- Το επίπεδο αποθήκευσης
  - Το επίπεδο δεδομένων είναι γενικά υπεύθυνο και για τη διατήρηση της συνέπειας των δεδομένων σε σχέση με διαφορετικές εφαρμογές
  - Τα δεδομένα είναι οργανωμένα ανεξάρτητα από τις εφαρμογές, με τέτοιο τρόπο ώστε οι αλλαγές σε αυτή την οργάνωση να μην επηρρεάζουν τις εφαρμογές, αλλά ούτε οι εφαρμογές να επηρρεάζουν την οργάνωση των δεδομένων
  - Στις περιπτώσεις εκείνες που οι λειτουργίες στα δεδομένα εκφράζονται ευκολότερα με χειρισμούς αντικειμένων, είναι λογικό το επίπεδο δεδομένων να υλοποιείται με μία αντικειμενοστραφή βάση δεδομένων
    - Μία τέτοια βάση διατηρεί αποθηκευμένη την υλοποίηση των λειτουργιών που μπορούν να πραγματοποιούνται σε αυτά τα αντικείμενα
    - Συνεπώς, ένα μέρος των λειτουργικών δυνατοτήτωνπου βρίσκονται στο επίπεδο επεξεργασίας μετακινείται τώρα στο επίπεδο δεδομένων

Υλοποίηση Μοντέλου Π-Ε

- Το κλασικό μοντέλο πελάτη-εξυπηρετητή αποτελείται από δύο λογικές οντότητες, μία για κάθε ρόλο, οι οποίες συνήθως βρίσκονται και σε διαφορετικές μηχανές
  - Ένα μηχάνημα πελάτη (διασύνδεση)
  - Ένα μηχάνημα διακομιστή (επεξεργασία, δεδομένα)
  - Η πιο απλή αρχιτεκτονική είναι να διατηρήσουμε μόνο το επίπεδο διεπαφής στον πελάτη και να μεταφέρουμε στον εξυπηρετητή τα επίπεδα επεξεργασίας και αποθήκευσης
  - Το πρόβλημα αυτού του τρόπου οργάνωσης είναι ότι δεν είναι πραγματικά κατανεμημένος

Πολυστρωματικές Αρχιτεκτονικές

- Διστρωματική αρχιτεκτονική από φυσική άποψη (physically two-tiered architecture)
  - Όταν το μοντέλο αυτό επεκτείνεται σε τρία λογικά επίπεδα, η κατανομή των επιπέδων σε μηχανές μπορεί να γίνει με διάφορους τρόπους
  - Μία λύση είναι να μοιράσουμε το επίπεδο διεπαφής ανάμεσα στον πελάτη και τον εξυπηρετητή, έτσι ώστε ο έλεγχος του τερματικού να γίνεται στον πελάτη αλλά η παρουσίαση δεδομένων και ο έλεγχος της εφαρμογής να ελέγχονται από τον εξυπηρετητή
    - Έτσι ο πελάτης είναι απόλυτα τυποποιημένος, ανεξάρτητα από τις εφαρμογές, αφού αυτές ουσιαστικά υλοποιούνται στον εξυπηρετητή
    - Για παράδειγμα, μπορεί να είναι ένας browser, με αποτέλεσμα οι εφαρμογές να μην απαιτούν πρόσθετες ενέργειες εγκατάστασης, διαχείρισης και συντήρησης

Πολυστρωματικές Αρχιτεκτονικές

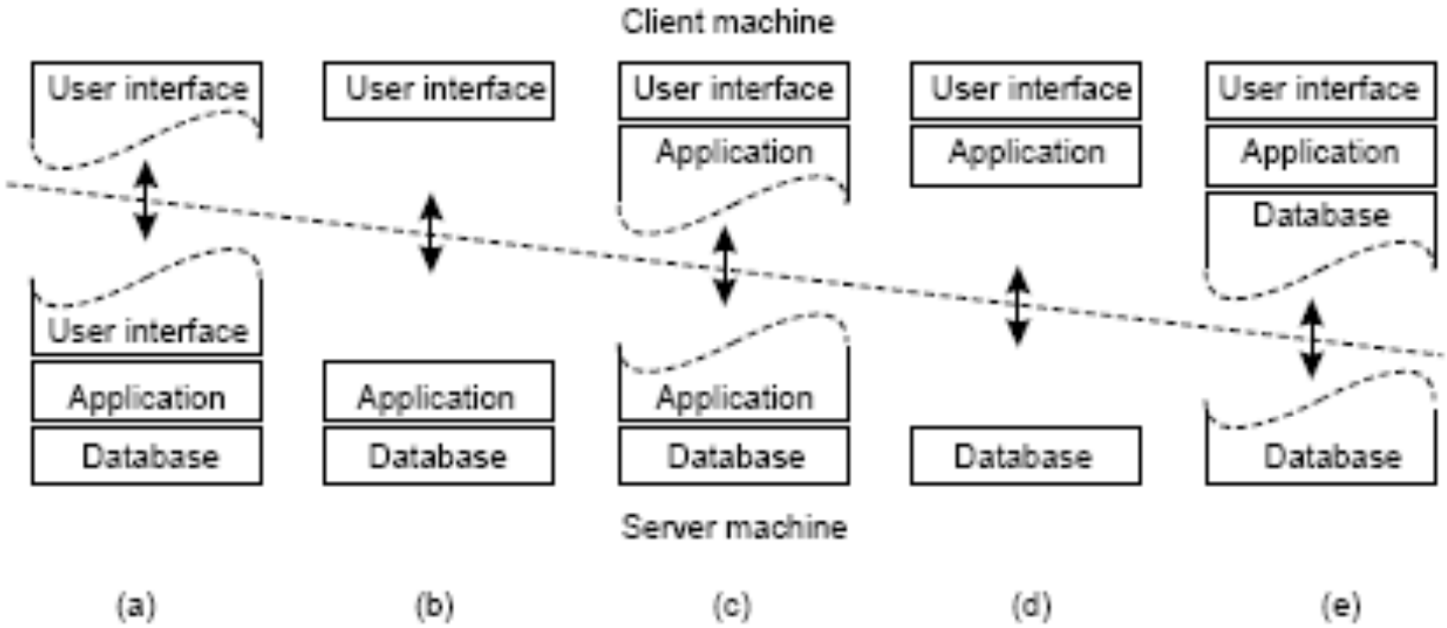


Fig. 1-29. Alternative client-server organizations (a)-(e).

Πολυστρωματικές Αρχιτεκτονικές

- Διστρωματική αρχιτεκτονική από φυσική άποψη (physically two-tiered architecture)
  - Κινούμενοι προς την αντίθετη κατεύθυνση, μπορούμε να μεταφέρουμε μέρος της επεξεργασίας στον πελάτη
    - Έτσι ώστε να γίνεται έλεγχος της εισόδου και κάποια επεξεργασία των αποτελεσμάτων τοπικά χωρίς να αναμιγνύεται ο εξυπηρετητής
    - Για παράδειγμα, ο πελάτης μπορεί να ελέγχει την εγκυρότητα των δεδομένων εισόδου σε πεδία ημερομηνίας
  - Μπορούμε να πάμε ακόμη παραπέρα, μεταφέροντας μέρος και του επιπέδου αποθήκευσης στον πελάτη
    - Για παράδειγμα, ένας browser σταδιακά δημιουργεί μία τοπική βάση στο δίσκο με ιστοσελίδες από το Internet έτσι ώστε να επιταχύνει την περιήγηση, ουσιαστικά αναλαμβάνοντας ένα μέρος του επιπέδου αποθήκευσης μέσω της τοπικής βάσης

Πολυστρωματικές Αρχιτεκτονικές

- Τριτρωματική αρχιτεκτονική από φυσική άποψη (physically three-tiered architecture)
  - Ο διακομιστής μπορεί να ενεργεί και σαν πελάτης
  - Τα προγράμματα που αποτελούν μέρος του επιπέδου επεξεργασίας βρίσκονται σε ξεχωριστό διακομιστή, αλλά υπάρχει επίσης δυνατότητα να είναι κατά ένα μέρος κατανεμημένα μεταξύ των μηχανημάτων πελάτη και διακομιστή
  - Παράδειγμα: επεξεργασία συναλλαγών
    - Μία ξεχωριστή διεργασία που ονομάζεται ελεγκτής συναλλαγών συντονίζει όλες τις συναλλαγές σε ενδεχομένως διαφορετικούς διακομιστές δεδομένων

Πολυστρωματικές Αρχιτεκτονικές

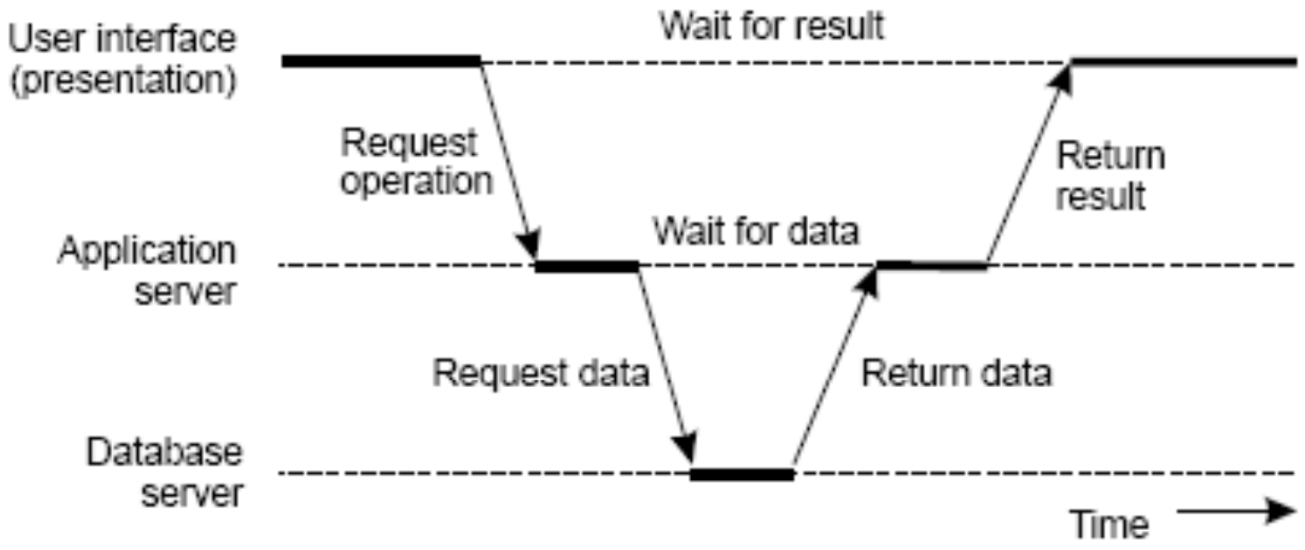


Fig. 1-30. An example of a server acting as client.

Υλοποίηση Μοντέλου Π-Ε

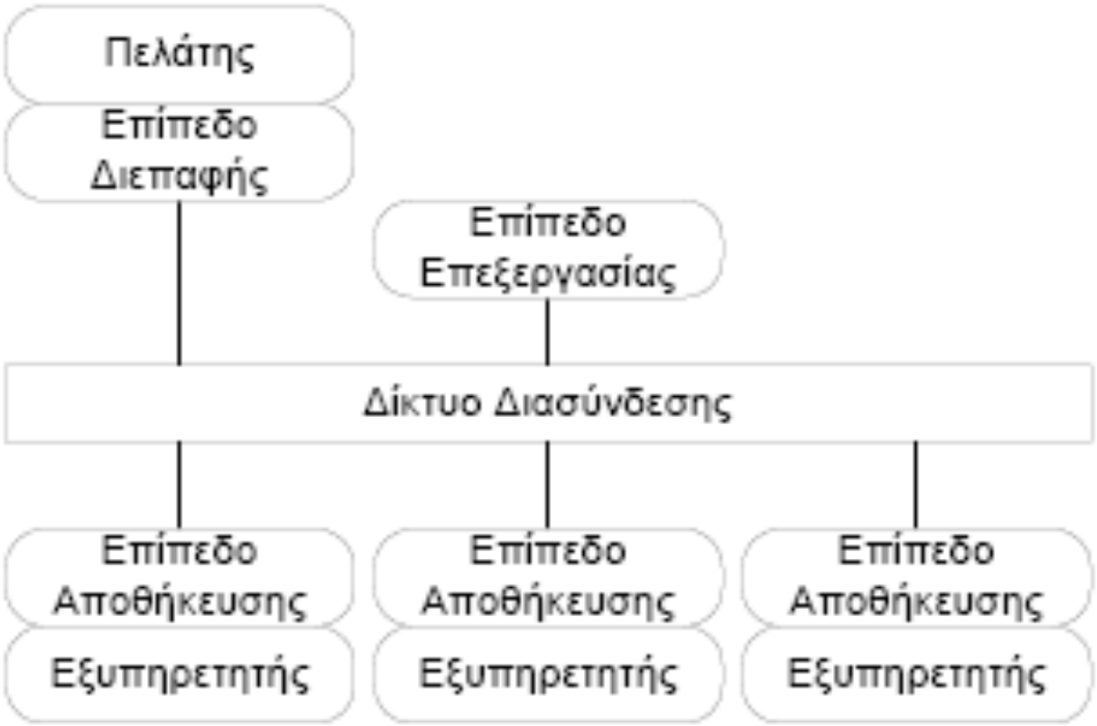
- Σε ένα κατανεμημένο σύστημα μεγάλης κλίμακας, εκτός από το ότι οι εφαρμογές μπορεί να κατανέμονται σε λογικά επίπεδα (διεπαφής, επεξεργασίας και αποθήκευσης), και τα ίδια τα επίπεδα επεξεργασίας και αποθήκευσης μπορεί να είναι κατανεμημένα

- *Κατακόρυφη κατανομή (vertical distribution)*
  - Η κατανομή διαφορετικών λογικά καθηκόντων σε διαφορετικές μηχανές, όπως στα μοντέλα τριών επιπέδων
  - Η κατανεμημένη επεξεργασία ισοδυναμεί με την οργάνωση μιας εφαρμογής Π-Ε με πολυστρωματική αρχιτεκτονική

Υλοποίηση Μοντέλου Π-Ε

- *Οριζόντια κατανομή (horizontal distribution)*
  - Η κατανομή ενός καθήκοντος σε πολλαπλές μηχανές
    - Ένας πελάτης ή διακομιστής μπορεί να υποδιαιρείται από φυσική άποψη σε λογικά ισοδύναμα μέρη, αλλά το κάθε μέρος δουλεύει πάνω στο δικό του μερίδιο του πλήρως συνόλου δεδομένων, ώστε να μοιράζεται ο φόρτος
  - Παράδειγμα 1: στο Σχήμα φαίνεται μία περίπτωση οριζόντια κατανομής του επιπέδου αποθήκευσης σε πολλούς εξυπηρετητές, πέρα από την κατακόρυφη κατανομή του μοντέλου πελάτη-εξυπηρετητή σε τρία επίπεδα
  - Παράδειγμα 2: Η οριζόντια κατανομή χρησιμοποιείται ευρύτατα στους εξυπηρετητές ιστοσελίδων, στους οποίους συνδυάζονται πολλαπλά αντίγραφα ενός εξυπηρετητή με υψηλό φόρτο στο επίπεδο αποθήκευσης με έναν κεντρικό δρομολογητή στο επίπεδο επεξεργασίας για να κατανέμονται διαφανώς τα αιτήματα των πελατών στους διαθέσιμους εξυπηρετητές

Υλοποίηση Μοντέλου Π-Ε



Σχήμα 1.7. Κατακόρυφη και οριζόντια κατανομή επεξεργασίας.

Υλοποίηση Μοντέλου Π-Ε

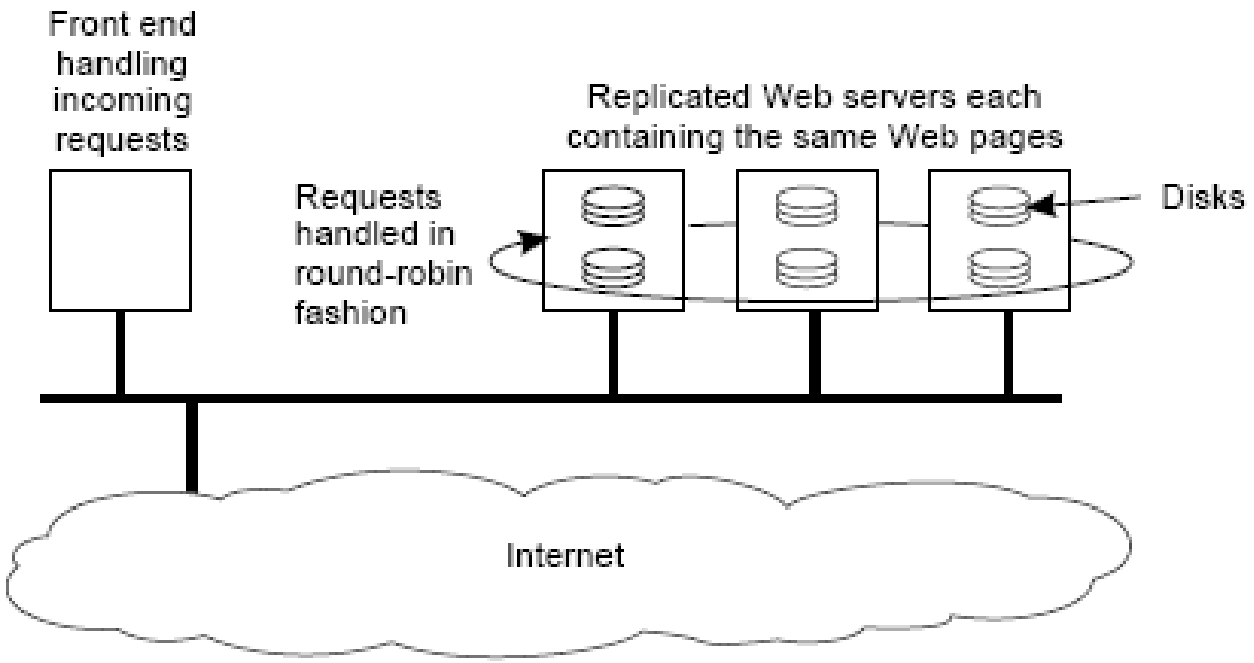


Fig. 1-31. An example of horizontal distribution of a Web service.

Υλοποίηση Μοντέλου Π-Ε

- *Οριζόντια κατανομή (horizontal distribution)*
  - Οι πελάτες μπορούν να είναι και αυτοί κατανεμημένοι
  - Για συνεργατικές εφαρμογές είναι δυνατό να μην υπάρχει καν διακομιστής
  - Τότε μιλάμε για ομότιμη κατανομή (peer-to-peer distribution)
  - Οι πελάτες εκτελούν την ίδια εφαρμογή για να ξεκινήσουν μία σύνοδο
  - Αργότερα ένας τρίτος πελάτης μπορεί να έρθει σε επαφή με οποινδήποτε από τους δύο, και στη συνέχεια να ξεκινήσει και αυτός το ίδιο λογισμικό εφαρμογής