Καστίδου Γεωργία Α.Μ. 42
Πέτσιος Κων/νος - Στέφανος Α.Μ. 47

Ιωάννινα, 17 Μαρτίου 2003

Κατανεμημένα Συστήματα
Εργασία 2

<u>Άσκηση 2.3</u> A reliable multicast service allows a sender to reliably pass messages to a collection of receivers. Does such a service belong to a middleware layer, or should it be part of a lower - level.

Οι διαδικασίες multicast πραγματοποιούνται με την βοήθεια του TCP το οποίο ανήκει στο middleware layer. Το IP4 έχει διαδικασίες για την απευθείας επικοινωνία δύο κόμβων αλλά και ενσωματωμένες broadcast διαδικασίες (δηλαδή εκπομπή σε ολόκληρα υποδίκτυα) τις οποίες μπορεί να εκμεταλλευτεί ο προγραμματιστής (application layer), έτσι ώστε να πετυχαίνει αποδοτικές multicast λειτουργίες. Έτσι φαίνεται το lower επίπεδο να μην συμμετέχει ενεργά, δηλαδή να μην διαφοροποιείται από ότι σε μία συνηθισμένη εφαρμογή, στην multicast λειτουργία.
Για αυτή την ιδιαίτερη μορφή επικοινωνίας multicast έχει γίνει ειδική μελέτη στο IPv6 στο οποίο πλέον όλες οι λειτουργίες του multicast πάνε στο επίπεδο δικτύου.

<u>Άσκηση 2.6</u> One way to handle parameter conversion in RPC systems is to have each machine send parameters in its native representation, with the other one doing the translation, if need be. The native system could be indicated by a code in the first byte. However, since locating the first word is precisely the problem, can this actually work?

Προσθέτουμε στην αρχή του μηνύματος ένα byte τέτοιο ώστε είτε μεταφραστεί σε little είτε σε big endian να έχει το ίδιο περιεχόμενο. Οπότε όταν ο παραλήπτης πάρει το μήνυμα κοιτάει το πρώτο byte και με βάση αυτό καταλαβαίνει αν χρησιμοποεί ο αποστολέας big είτε little endian.

<u>Άσκηση 2.8</u> Instead of letting a server register itself with a daemon as is done in DCE, we could also choose to always assign in the same endpoint. That endpoint can then be used in references to objects in the server's address space. What is the main drawback of this scheme?

Το κύριο μειονέκτημα της χρήσης του endpoint είναι ότι στην περίπτωση που για κάποιο λόγο μια άλλη εφαρμογή Β πιάσει το endpoint που χρησιμοποιεί μια εφαρμογή Α τότε θα χρειαστεί η Α να αλλάξει endpoint και να ειδοποιήσει όλες τις άλλες εφαρμογές που συμμετέχουν σε κάποια επικοινωνία με την Α. Το πρόβλημα αυτό γίνεται μεγαλύτερο στην περίπτωση που η Α δεν γνωρίζει τις άλλες αυτές εφαρμογές. Α και στην περίπτωση που η εφαρμογή που Β έχει κακές προθέσεις και θελήσει να υποκριθεί ότι είναι η Α. Τα παραπάνω προβλήματα λύνονται με τη χρήση του daemon.

Ουσιαστικά ένα endpoint δεν μας προσφέρει έναν αποδοτικό και καλά προστατευμένο κατανεμημένο μηχανισμό επικοινωνίας.

<u>Άσκηση 2.9</u> Give an example implementation of an object reference that allows a client to bind to a transient remote object.

Ένα παράδειγμα υλοποίησης transient remote object είναι οι κινητοί πράκτορες στη γενική τους περίπτωση. Αν θελήσουμε να στείλουμε ένα μήνυμα σε έναν κινητό πράκτορα και ο κινητός πράκτορας έχει μετακινηθεί το μήνυμα θα χαθεί.

¨Ενα παράδειγμα κινητών πρακτόρων που δουλεύουν με τον παραπάνω τρόπο είναι οι Aglets της IBM:

```
/*****************************************************************************/
public AgletProxy find_Receiver_proxy(URL Receiver_URL,AgletID Receiver_ID) {
    try {
// Στο σημείο βρίσκουμε το proxy του πράκτορα που βρίσκεται στη θέση Receiver_URL και έχει
// ταυτότητα ίση με  Receiver _ID
        return getAgletContext().getAgletProxy(Receiver _URL, Receiver _ID);
    } catch (Exception e) {
      System.out.println(e);
     }
    return null;
  }

  public void send_hello_message() {
     try {
// Στο σημείο βρίσκουμε το proxy του Receiver
          AgletProxy Receiver_proxy=find_Receiver_proxy(Receiver_URL Receiver_ID);
// Στο σημείο αυτό μέσω του proxy γίνεται η αποστολή μηνύματος στον κινητό πράκτορα Receiver
        Receiver_proxy.sendMessage(new Message("Hello"));
       } catch (Exception e) {
// Σε περίπτωση που ο Receiver έχει μετακινηθεί το μήνυμα δεν θα παραδοθεί και εκκινηθεί μία
// διαδικασία exception
           System.out.println(e);
       }
     return;
  }
/*****************************************************************************/
```

Άσκηση 2.15 Suppose that you could make use of only transient asynchronous communication primitives, including only an asynchronous receive primitive. How would you implement primitives for transient synchronous communication?

Αυτό που μπορεί κάποιος να κάνει είναι ο προγραματιστής να φροντίσει για κάθε αποστολή ενός μηνύματος ο δέκτης να απαντά με ένα μήνυμα επιβεβαίωσης λήψης, ενώ ο αποστολέας να μπλοκάρει τη λειτουργία του μέχρι να έρθει το παραπάνω μήνυμα. Με τον όρο «μπλοκάρει» εννοούμε ο αποστολές να εκτελεί ένα είδος while όπου ανα κάποιο χρονικό διάστημα t να ελέγχει αν έχει φτάσει το μήνυμα παράδοσης του αρχικού μηνύματος από τον παραλήπτη του.

Με τον τρόπο αυτό ουσιαστικά εξομοιώνουμε ένα (receipt-based) transient synchronous μοντέλο με την χρήση transient asynchronous.

Άσκηση 2.22 Explain why transient synchronous communications has inherent scalability problems and how these could be solved.

Ένα από τα προβλήμα είναι η ανίχνευση σφάλματος – αποτυχίας (failure) η οποία μπορεί να οφείλεται είτε στην εσφαλμένη επικοινωνία είτε στην αποτυχία του παραλήπτη. Στις παραπάνω περιπτώσεις ο αποστολέας μπλοκάρει χωρίς να μπορεί να επανέλθει. Κάποιες υλοποιήσεις που βασίζονται σε time outs για την ανίχνευση σφαλμάτων αντιμετωπίζουν το παραπάνω πρόβλημα αλλά εισάγουν νέα προβλήματα όπως για παράδειγμα η «σωστή» επιλογή του χρονικού διαστήματος που θα επιλεχθεί για το "time out", ή το πρόβλημα της επιλογής μεταβλητού time out η τιμή του οποίου εξαρτάται από το μήνυμα που θέλει ο αποστολέας να στείλει.

Άσκηση 2.24 How could you guarantee a maximum end-to-end delay when a collection of computers is organized in a (logical or physical) ring?

Στην περίπτωση αυτή ως χείριστο delay είναι το μισό πλήθος (n/2) των κόμβων που συμμετέχουν στον δακτύλιο. Αυτό γίνεται με την προϋπόθεση ότι η επικοινωνία του κάθε κόμβου με τους γείτονές του είναι αμφίδρομη, οπότε τα μηνύματα στέλνονται προς τις δύο κατευθύνσεις του δακτυλίου και μπορούμε να δεσμεύουμε το μήνυμά μας να έχει "time to live" n/2 hops.

Στη περίπτωση που έχουμε που έχουμε επικοινωνία μιας κατεύθυνσης (one way) ως χείριστο delay είναι το n-1, και συμβαίνει όταν ένας κόμβος προσπαθεί να επικοινωνήσει με τον προηγούμενο του κόμβο.

Άσκηση 2.26 Imagine we have a token bucket specification where the maximum data unit size is 1000 bytes, the token bucket rate is 10 million bytes/sec, the token bucket size is 1 million bytes, and the maximum transmission rate is 50 million bytes/sec. How long can a burst of maximum speed last?

Τα μηνύματα χωρίζονται σε blocks των 1000bytes οπότε σε στιγμές με burst στο maximum speed ισχύει ότι:

Εισερχόμενα πακέτα: 50.000.000 / 1000 = 50.000 πακέτα σε κάθε δευτερόλεπτο.
Εξερχόμενα πακέτα: 10.000.000 / 1000 = 10.000 πακέτα σε κάθε δευτερόλεπτο.

Στην μνήμη έχουμε την δυνατότητα να 1.000.000 / 1000 = 1.000 πακέτα

Έτσι ο χρόνος που η μηχανή μας θα αντέξει στο burst αυτό είναι:

$$\frac{1000}{50.000 - 10.000} = \frac{1}{40}\text{sec}$$