Καρακασίδης Αλέξανδρος
Καστίδου Γεωργία
Παπαφώτη Μαρία
Πέτσιος Κων/νος – Στέφανος
Σαλτέας Καλογεράς Παναγιώτης

Remote Method Invocation (RMI)

Εισαγωγή

Η απομακρυσμένη επίκληση μεθόδων (RMI), επιτρέπει σε εφαρμογές την κλήση μεθόδων από απομακρυσμένα αντικείμενα, για το διαμοιρασμό πόρων και επεξεργαστικού φόρτου μεταξύ των συστημάτων. Σε αντίθεση με άλλα συστήματα απομακρυσμένης εκτέλεσης που επιτρέπουν μόνο τη μεταφορά απλών τύπων δεδομένων ή συγκεκριμένων δομών, το RMI επιτρέπει σε οποιοδήποτε αντικείμενο της Java να χρησιμοποιηθεί, ακόμα και αν ο εξυπηρέτης δε το έχει ξανασυναντήσει. Το RMI επιτρέπει τόσο στον πελάτη όσο και στον εξυπηρέτη να φορτώσουν δυναμικά νέους τύπους αντικειμένων.

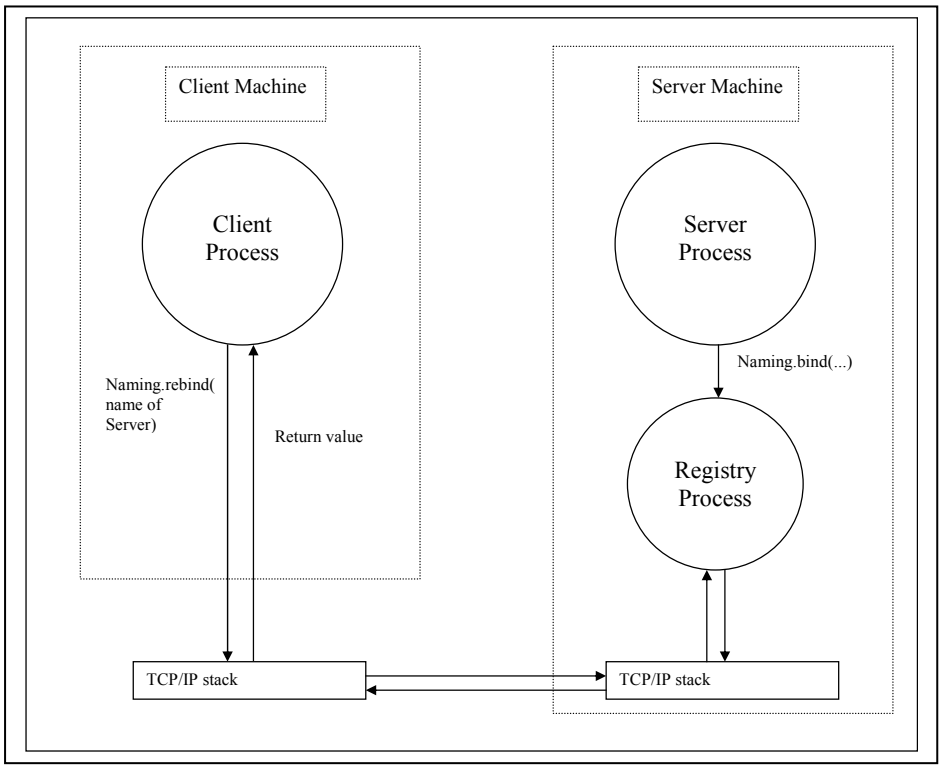
1. Γενικά

Οι εφαρμογές RMI συνήθως αποτελούνται από δύο ξεχωριστά προγράμματα: τον εξυπηρέτη (server) και τον πελάτη (client). Μια τυπική εφαρμογή εξυπηρέτη δημιουργεί μερικά απομακρυσμένα αντικείμενα, αναφορές σε αυτά ώστε να είναι προσβάσιμα, και περιμένει τους πελάτες να καλέσουν μεθόδους πάνω σε αυτά. Μια τυπική εφαρμογή πελάτη καλεί μεθόδους πάνω σε απομακρυσμένα αντικείμενα μέσω μιας απομακρυσμένης αναφοράς. Το RMI παρέχει το μηχανισμό επικοινωνίας ανάμεσα στον εξυπηρέτη και στον πελάτη. Τέτοιες εφαρμογές αναφέρονται και ως εφαρμογές κατανεμημένων αντικειμένων (*distributed object applications*).

Οι εφαρμογές κατανεμημένων αντικειμένων:

- Χρειάζεται να εντοπίσουν τα κατανεμημένα αντικείμενα: Οι εφαρμογές είναι δυνατόν να χρησιμοποιήσουν έναν από τους δυο μηχανισμούς που αναφέρονται παρακάτω για να αποκτήσουν αναφορές σε απομακρυσμένα αντικείμενα.
- Μπορούν να καταγράφουν τα απομακρυσμένα αντικείμενα με τη βοήθεια της RMI's simple naming facility
- Μπορούν να θεωρήσουν το πέρασμα και την επιστροφή των αναφορών σε απομακρυσμένα αντικείμενα ως ενσωματωμένη λειτουργία της εφαρμογής.

- Επικοινωνούν με απομακρυσμένα αντικείμενα: Το RMI διαχειρίζεται τις λεπτομέρειες της επικοινωνίας. Όσον αφορά τον προγραμματιστή, αυτός κατανοεί την απομακρυσμένη επικοινωνία σαν μια απλή κλήση μιας τοπικής μεθόδου της Java.
- Φορτώνουν τον κώδικα των απομακρυσμένων αντικειμένων που εκτελούνται. Επειδή το RMI επιτρέπει στον καλούντα να περνάει αντικείμενα σε άλλα απομακρυσμένα αντικείμενα, παρέχει τον απαιτούμενο μηχανισμό φόρτωσης του κώδικα ενός αντικειμένου καθώς και μηχανισμό για τη μεταφορά των δεδομένων του.



Σχήμα 1

Στην κατανεμημένη πλατφόρμα που μας παρέχει η Java, ένα απομακρυσμένο αντικείμενο είναι δυνατό να το καλέσουμε με τη χρήση απομακρυσμένων κλήσεων (μεθόδους) ανάμεσα σε διαφορετικές JVM (που μπορεί να είναι σε διαφορετικούς κόμβους – υπολογιστές).

Οι απομακρυσμένες κλήσεις περιγράφονται σε ένα ή περισσότερα απομακρυσμένα interfaces, τα οποία είναι γραμμένα σε Java και ορίζουν την δήλωση των κλάσεων και των μεθόδων που περιέχονται σε αυτές.

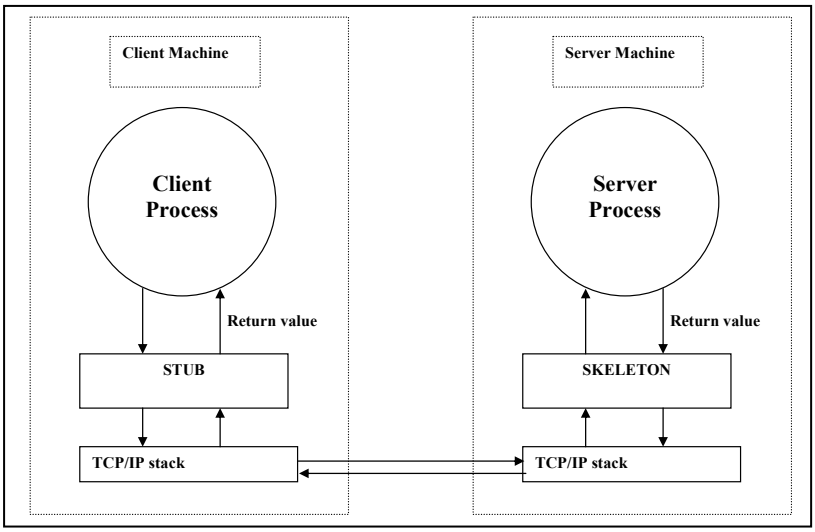
2. Επικλήσεις Απομακρυσμένων Μεθόδων

Ένα remote object είναι ένα αντικείμενο, το οποίο βρίσκεται σε μία απομακρυσμένη μηχανή, αλλά μπορούμε να το χειριστούμε σαν να βρισκόταν στην τοπική μηχανή

μας. Έτσι όταν γίνει κλήση μιας απομακρυσμένης κλάσης, η τοπική μηχανή το ανιχνεύει και αναλαμβάνει να πραγματοποιήσει τις αντίστοιχες κλήσεις που βρίσκονται στην απομακρυσμένη μηχανή με πλήρη διαφάνεια προς τον χρήστη. Η κλήση της μεθόδου, η οποία ανήκει σε ένα απομακρυσμένο αντικείμενο ονομάζεται: *remote method invocation (RMI).*

Το απομακρυσμένο αντικείμενο προσφέρει τις μεθόδους του σαν να είναι υπηρεσίες. Οι πελάτες είναι οι διεργασίες οι οποίες εκκινούν την εκτέλεση των μεθόδων αυτών.

Στη Java όταν εκτελούμε RMI κλήσεις, στις οποίες ο πελάτης και ο εξυπηρέτης βρίσκονται σε διαφορετικές JVM, τότε γίνεται μία κανονική ανταλλαγή μηνυμάτων ανάμεσα στις δύο μηχανές. Οι παράμετροι της απομακρυσμένης μεθόδου στέλνονται με την μορφή ενός μηνύματος, τη δημιουργία και την αποστολή του οποίου αναλαμβάνει να κάνει μία *stub* κλάση που βρίσκεται στον πελάτη. Την κλάση stub ο πελάτης την κατεβάζει από τον εξυπηρέτη. Το μήνυμα αυτό το λαμβάνει ο εξυπηρέτης στον οποίο μία *skeleton* κλάση αναλαμβάνει να κάνει την αποκωδικοποίηση των παραμέτρων καθώς επίσης να κάνει και την άμεση κλήση στον εξυπηρέτη.



Σχήμα 2

3. Stubs και Skeletons

Το RMI χρησιμοποιεί ένα συγκεκριμένο μηχανισμό (που υπάρχει και στα συστήματα RPC) για την επικοινωνία με τα απομακρυσμένα αντικείμενα: τα stubs και τα skeletons. Για ένα απομακρυσμένο αντικείμενο το stub λειτουργεί ως ο τοπικός αντιπρόσωπος (ή proxy) του απομακρυσμένου αντικειμένου. Ο καλών παρεμβάλλει μία μέθοδο στο τοπικό stub το οποίο είναι υπεύθυνο για τη διενέργεια των κλήσεων των μεθόδων στο απομακρυσμένο αντικείμενο. Στο RMI συγκεκριμένα, οι μέθοδοι που υπάρχουν στο stub είναι αυτές που υλοποιούνται από το απομακρυσμένο αντικείμενο.

Κατά την επίκληση μίας μεθόδου του stub, γίνονται τα παρακάτω:

- Εκκίνηση μίας σύνδεσης με την απομακρυσμένη JVM που περιέχει το

απομακρυσμένο αντικείμενο.
- Εγγραφή και μετάδοση των παραμέτρων στην απομακρυσμένη JVM.
- Αναμονή του αποτλέσματος της απομακρυσμένης επίκλησης της μεθόδου.
- Ανάγνωση της απάντησης ή της εξαίρεσης που επιστρέφεται από την απομακρυσμένη επίκληση.
- Επιστροφή της τιμής στον καλώντα.

Το stub ουσιαστικά κρύβει τις επικοινωνίες του επιπέδου δικτύου, ώστε να παρέχει ένα απλό μηχανισμό επίκλησης μεθόδων.

Στην απομακρυσμένη μηχανή, το απομακρυσμένο αντικείμενο έχει ένα αντίστοιχο skeleton. Αυτό είναι υπεύθυνο για τη διαβίβαση της κλίσης από το skeleton στην υλοποίηση του αντικειμένου. Όταν το skeleton λαμβάνει ένα μήνυμα κάνει τα εξής:
- Διαβάζει τις παραμέτρους για την απομακρυσμένη μέθοδο
- Κάνει επίκληση της μεθόδου στο πραγματικό απομακρυσμένο αντικείμενο
- Γράφει και μεταδίδει το αποτέλεσμα (τιμή ή εξαίρεση) στον πελάτη.

4. Εύρεση Απομακρυσμένων Αντικειμένων – Χρήση του Registry

Πριν ένας πελάτης χρησιμοποιήσει ένα απομακρυσμένο αντικείμενο θα πρέπει να μπορεί να το βρεί. Το RMI προσφέρει ένα απλό σχήμα ονοματοδοσίας, στο οποίο ένα απομακρυσμένο αντικείμενο δίνει στον εαυτό του ένα όνομα όταν τρέχει για πρώτη φορά. Στη συνέχεια καταχωρείται στο RMI registry με μια διαδικασία εγγραφής. Το registry συνδέει το όνομα του αντικειμένου (όχι το όνομα της κλάσης) και το ίδιο το αντικείμενο.

Στη Java, όταν ένα απομακρυσμένο αντικείμενο εγγράφεται στο registry μίας συγκεκριμένης μηχανής, συνδέται με ένα αντικείμενο ονοματοδοσίας. Το αντικείμενο ονοματοδοσίας είναι ένα παγκόσμιο αντικείμενο, αντίστοιχα με το αντικείμενο συστήματος (System Object) στη Java.

Αν ένα πελάτης θέλει να χρησιμοποιήσει ένα αντικείμενο, το οποίο βρίσκεται σε έναν απομακρυσμένο κόμβο (έστω κόμβος Α), κάνει μία αναζήτηση στο Registry του κόμβου Α. Στη συνέχεια, μπορεί να χρησιμοποιήσει το αποτέλεσμα της αναζήτησηςγια να συνδεθεί με το απομακρυσμένο αντικείμενο, και να παρεμβάλλει τις μεθόδους του.

5. Δυναμική φόρτωση κλάσεων

Στο προγραμματιστικό περιβάλλον του RMI ο προγραμματιστής έχει την δυνατότητα να μην είναι δεσμευμένος εξ' αρχής για τις μεθόδους που θα καλέσει. Δηλαδή, μας παρέχεται η δυνατότητα, ο πελάτης να στείλει στον εξυπηρέτη τη μέθοδο που επιθυμεί να εκτελεστεί. Για να γίνει αυτό θα πρέπει να υπάρχει και στον πελάτη εν λειτουργία εξυπηρέτης Web.

6. Κανόνες για τον εξυπηρέτη και τον πελάτη

6.1 Το πρόγραμμα του εξυπηρέτη

1. Ανάγνωση πολιτικής ασφαλείας (όταν δεν έχουμε HTTP μπορούμε να φιλτράρουμε τους πελάτες που θα έχουν πρόσβαση)
2. Σύνδεση με το Registry
3. Αναμονή για rmi requests.

Ο κώδικάς του εξυπηρέτη επίσης ορίζει έναν κατασκευαστή και κάποιες μεθόδους.

6.2 Το πρόγραμμα του πελάτη

1. Ανάγνωση πολιτικής ασφαλείας (αν χρειάζεται).
2. Εντοπισμός και σύνδεση με τον εξυπηρέτη.
3. Πραγματοποίηση απομακρυσμένων κλήσεων.

7. Δημιουργία Remote Method Invocations στη Java

Τα βήματα που πρέπει να κάνει κάποιος για να χρησιμοποιήσει το RMI.

7.1 Στον εξυπηρέτη:

1. Μεταγλώτισση του εξυπηρέτη:
π.χ. javac server.java

2. Δημιουργία των Skeleton και Stub
π.χ. rmic server

3. Εκκίνηση του registry (την πρώτη φορά)
rmiregistry &

4. Εκτέλεση του εξυπηρέτη
π.χ. java server *hostname*

Παράδειγμα Εξυπηρέτη:

```
public class server  extends UnicastRemoteObject  implements Echo {
        public server() throws RemoteException {
                        super();
        }
        public String call(String message) throws RemoteException        {
```

```
                return  "From EchoImpl: Thanks for your message: " + message;
        }
        public static void main(String args[])          {
                // Create and install the security manager
                System.setSecurityManager(new RMISecurityManager());
                try {
                    // create an EchoImpl server
                    server echo = new server();
                    String url = "//" + args[0] + ""/~username/rmi/Echo ";
                    // bind the url to echo
                    Naming.rebind(url, echo);
                } catch (Exception e) {
                      System.out.println( e.getMessage() );
                }
        }
}
```

7.2 Στον πελάτη:

1. Μεταγλώτισση του πελάτη:
   π.χ. javac client.java

2. Εκτέλεση του πελάτη με καθορισμό της θέσης του stub
   π.χ. java -Djava.rmi.server.codebase = http://*hostname*/~*username*/*directory*/ *clientclass hostname method parameters*

Στην εφαρμογή μας, ο εξυπηρέτης τρέχει στον zeus.cs.uoi.gr:
java server zeus.cs.uoi.gr

Ο πελάτης τρέχει σε οποιοδήποτε άλλο μηχάνημα. Ο πελάτης μπορεί:
I) Να ψηφίσει
   java -Djava.rmi.server.codebase = http://*zeus.cs.uoi.gr*/~stefanos/rmi/ client vote όνομα_ψηφοφόρου όνομα υποψηφίου

II) Να μάθει αποτελέσματα ψηφοφορίας είτε κάποιου συγκεκριμένου υποψήφιου είτε τα γενικά αποτελέσματα
   java -Djava.rmi.server.codebase = http://*zeus.cs.uoi.gr*/~stefanos/rmi/ client results όνομα υποψηφίου
   ή
   java -Djava.rmi.server.codebase = http://*zeus.cs.uoi.gr*/~stefanos/rmi/ client results all

III) Να δει τη λίστα των υποψυφίων
   java -Djava.rmi.server.codebase = http://*zeus.cs.uoi.gr*/~stefanos/rmi/ client display_candidates

Παράδειγμα Πελάτη :

```
public class Client {
  public static void main(String args[])       {
    // Create and install the security manager
    System.setSecurityManager(new RMISecurityManager());
    try {
      String url = "//" + args[0] + "/~username/rmi/Echo";
      // lookup Echo server
      Echo echo = (Echo) java.rmi.Naming.lookup(url);
      // call remote method
      String message = echo.vote(args[1],args[1]);
      // print message returned from server
      System.out.print("Message from Echo Server: " + message);
    } catch (Exception e) {
      System.out.println(e.getMessage());
    }
}
}
```

8. Περιορισμοί και Ζητήματα Επιδόσεων

8.1 Απόδοση και περιορισμοί του RMI

Οι κλήσεις με την χρήση HTTP είναι τουλάχιστον μία τάξη μεγέθους αργότερες από αυτές που κάνουν απευθείας χρήση sockets, χωρίς δηλαδή να λαμβάνουμε υπόψη τις καθυστερήσεις που οφείλονται στα proxys. Επίσης η χρήση HTTP είναι προβληματική επειδή εκ φύσεως πρόκειται για μονόδρομη επικοινωνία. Έτσι όταν επιθυμούμε αμφίδρομη επικοινωνία θα πρέπει να υπάρχει HTTP server και στον πελάτη αλλά και στον εξυπηρέτη και φυσικά εάν υπάρχουν firewalls να μην παρεμποδίζουν την μεταξύ τους επικοινωνία.

8.2 RMI μέσα από Firewalls με τη χρήση Proxies

Το επίπεδο μεταφοράς στο RMI κανονικά επιδιώκει να κάνει απ' ευθείας σύνδεση με τον αντίστοιχο host. Υπάρχουν όμως πολλά intranets τα οποία έχουν firewalls και δεν επιτρέπουν τέτοιου είδους απευθείας συνδέσεις. Το RMI για την επίλυση αυτών των προβλημάτων προσφέρει δύο εναλλακτικές μεθόδους επικοινωνίας των δύο άκρων οι οποίες βασίζονται σε μεθόδους tunneling και στην χρήση ενός ενδιάμεσου κόμβου με τον οποίο μπορούν να επικοινωνήσουν και οι δύο άκρες.

Αναφορές

1. http://www.javacoffeebreak.com/articles/javarmi/javarmi.html
2. http://java.sun.com/docs/books/tutorial/rmi/
3. Andrew S. Tanenbaum and Maarten Van Steen, "Distributed Systems: Principles and Paradigms", Prentice Hall, 2002