

---

# RMI: vzdialené volanie metód v Java

Róbert Novotný

(UINF/KOPR, 3. december 2013)

# Distribučnosť ~ výmena správ

---

Ako vymieňať  
správy medzi  
komponentami  
distribučovaného  
systému?



# RMI: zabudovaný spôsob v Jave

---

objekt

bajty

TCP  
drôt

# Remote Method Invocation

---

- transportná vrstva: **TCP**
- serializácia: **binárna**
- interop: len Java vs. Java

<http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp>

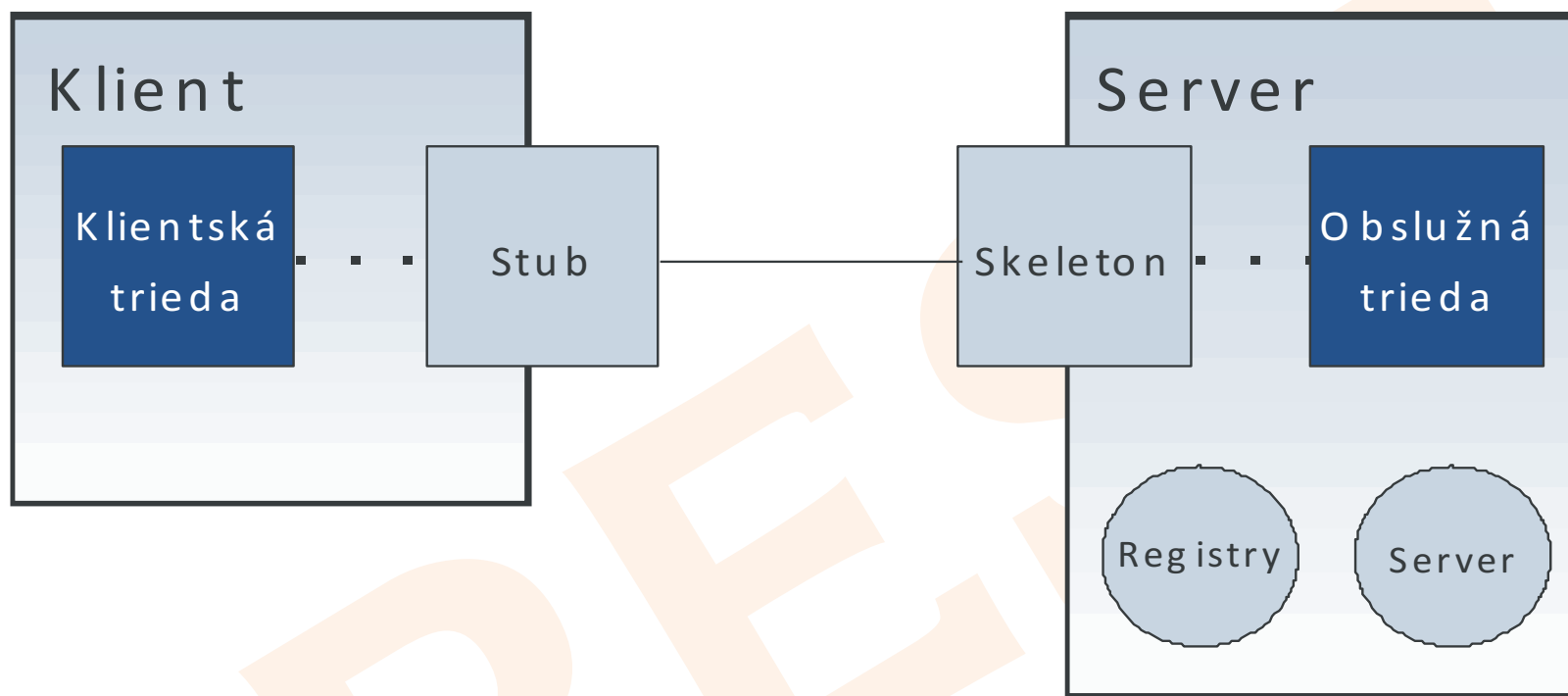
# RMI – ako vyzerá komunikácia?

---

- ako vyzerá typická správa?
- binárny neporiadok posielať cez TCP
- objekty sú serializované pomocou Java Serialization
  - java.io.**DataOutputStream**

<http://java.sun.com/javase/6/docs/platform/serialization/spec/serialTOC.html>

# Architektúra RMI



# Architektúra

---

- **vzdialený objekt** = obslužná trieda: metódy sú volané klientom.
- **registry**: register vzdialených objektov
- **server**: registruje vzdialené objekty
- **skeleton**: špinavá práca na serveri
  - sieťová komunikácia, (de)serializácia objektov
- **stub**: špinavá práca i klienta

# Vytvorenie servera

---

- vytvoriť vzdialený interface s metódami
  - **extends Remote**
  - každá metóda **throws RemoteException**
- vytvoriť implementačnú triedu
  - parametre a návratové hodnoty musia **implements Serializable**
- zaregistrovať ju v **Registry**



# Vytvorenie klienta

---

- získať objekt **Registry** z daného servera
- získať **vzdialený interfejs** (za ním je stub)
- na ňom volať metódy

---

# DEMO

[RMI služba + klient]




<http://ics.upjs.sk/~novotnyr/wiki/Java/RMI>



# Výhody a nevýhody

---

- ✓ jednoduché použitie, zabudované
- ✓ chýbajúce binárky možno dotiahnuť zo servera
  - ✗ pomerne náročná konfigurácia
  - ✗ bezpečnostné riziko
- ✗ firewally blokujú komunikáciu
  - ✓ možno tunelovať cez HTTP
  - ✗ náročná konfigurácia



Ak nepotrebujeme interop a vieme, že  
firewally nie sú problémom, je RMI  
najjednoduchšou cestou.

