05. Synchronní a asynchronní distribuované systémy, specifika návrhu a vlastností distribuovaných systémů, integrace externích datových zdrojů.(A4M33NMS)

Za distribuovaný lze v zásadě považovat každý systém, který pro své fungování **využívá více nezávislých výpočetních jednotek** (počítačů, procesorových jader).

Distribuované výpočty

= místo použití 1 superpočítače lze použít více běžných počítačů a výpočet paralelizovat
- jednotlivé uzly spolu komunikují např. pomocí RMI
- prvotní složitá úloha se pomocí **více-vláknové** technologie rozdělí na dílčí úlohy, které se rozdistribuují na další uzly
- mezivýsledky se následně složí dohromady

Serializace objektů

- lze objekt převést do binární nebo textové podoby a následně poslat po síti na vzdálený počítač
- tam se opět deserializuje a je možné k danému objektu přistupovat tak, jako by byl lokální
- proces serializace není závislý na OS ani technologii

Technologie

RMI – remote method invocation

- na serveru se nachází třída, která provádí výpočet
- třída má implementaci a rozhraní
- klient volá vzdálenou třídu pomocí RMI a potřebuje znát pouze její rozhraní
- JAVA RMI využívá Java serializaci pro přenos objektů

JMS – Java messice service

- Asynchronní
- Součástí Java EE
- Umožňuje loose-coupling, protože nevyžaduje přesnou znalost příjemce zprávy

Dva modely komunikace při použití JMS:

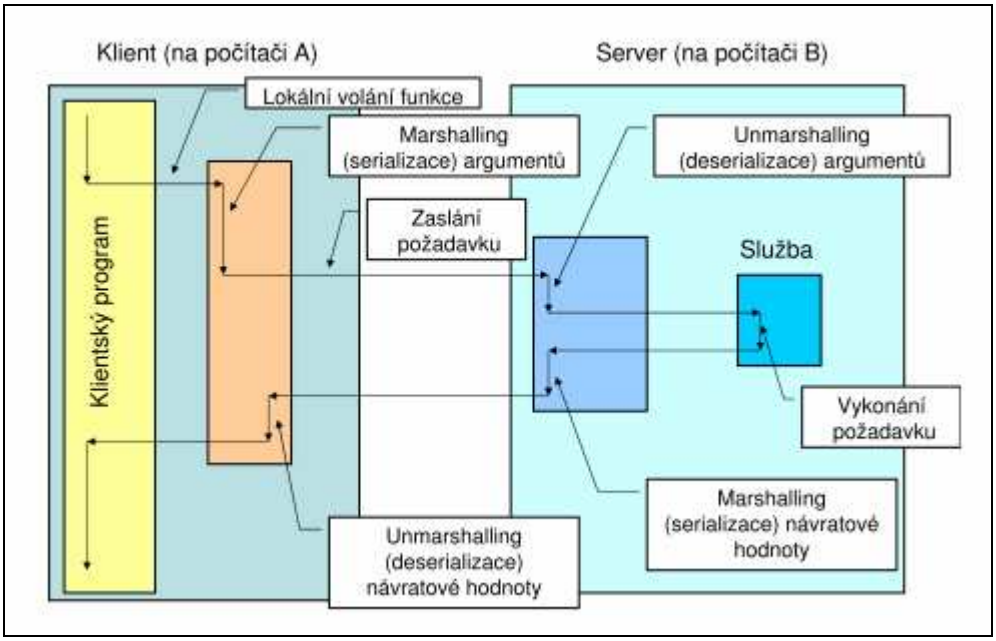
Point-to-point (queue)

- Odesílatel zná adresu příjemce (ten je jen jeden)
- Odeslaná zpráva se uloží do příjemcovy **fronty**
- Čili příjemce nemusí běžet, když odesílatel odesílá zprávu, stejně tak odesílatel nemusí běžet, když příjemce zpracovává zprávu
- Zpracování je příjemcem potvrzeno

Publish / Subscribe (topic)

- Odesílatel může zakládat různá **témata** (topic)
- Příjemci se **registrují k těmto tématům**
- Odesílatel označí zprávu tématem a ta následně dorazí všem registrovaným příjemcům
- Zprávu tedy obdrží 0 – n příjemců.
- Příjemce, který není aktuálně spuštěný, zprávu nezachytí (neuloží se v žádné frontě)

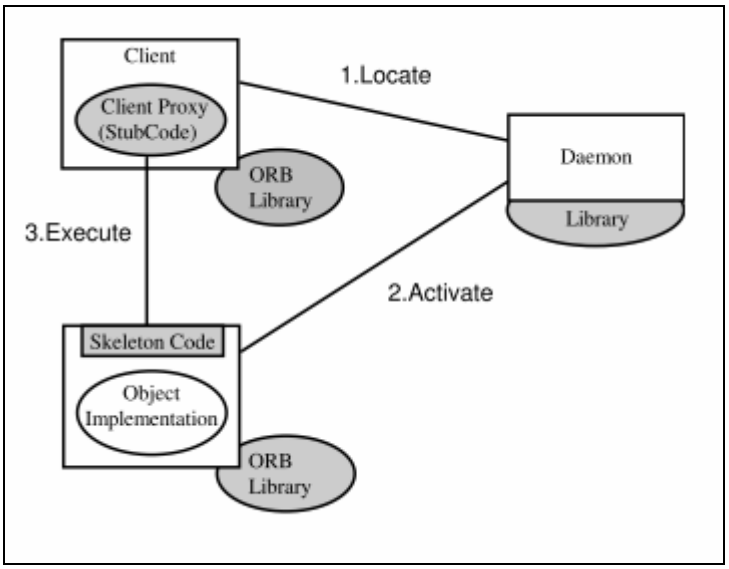
Remote Procedure Call (RPC)



- Synchronní
- Pomocí jazyka **IDL** (interface description language) lze **generovat automaticky kód** serverové i klientské části programu, který využívá RPC
- Často používaným nástrojem v jazyce C je RPCGEN.
- Jde o obecný princip, který je využit v mnoha podobách (CORBA, XML-RPC,SOAP,WCF a mnoha dalších)

CORBA – Common Object Request Broaker Architecture

- Standard definovaný skupinou OMG, který je jazykově i platformně nezávislý
- Se vzdálenými objekty aplikace komunikuje skrze **ORB** (Object Request Broker), což je **middleware**, který zajišťuje přenos objektů včetně serializace a deserializace. S objektem pak aplikace pracuje skrze interní Object Adapter.
- Struktura objektů je popsána pomocí **IDL**
- Objekty jsou přenášeny jako reference pomocí řetězce URI, zatímco data (integer, double..) jsou přenášena jako hodnoty.
- Výhoda je kontrola datových typů (Strong Data Typing), která je ale flexibilní (existuje i podpora zástupného typu „ANY“)
- **Nevýhody**: Běží na separátním TCP portu, který nemusí být vždy povolen na firewallech; Implementační problémy – neúplné, neefektivní implementace nebo komerční a drahé.



WebServices

- Pro volání vzdálených metod a přenos objektů využívá protokol SOAP založený na XML. Snadná integrace s http
- Jako IDL se používá formát WSDL
- V Java EE balíček JAX-RPC nebo JAX-WS

Integrace externích datových zdrojů

- ODBC – Open Database Connectivity
  - Standardní rozhraní pro přístup k DBMS, nezávislé na programovacím jazyku
  - Podporuje relační i nerelační databáze
- JDBC – Java Database Connectivity
  - Vrstva mezi Java aplikací a relační databází
  - Musí existovat JDBC ovladač, který převádí požadavky z Javy do podoby srozumitelné pro daný DBMS
  - Existuje JDBC-ODBC driver, který umožňuje přístup k jakékoli databázi podporující ODBC
- LDAP
  - Protokol pro čtení a editaci adresářů (organizovaných množin záznamů)
  - Často se používá jako zdroj jednotné autentizace uživatele (ověření proti LDAP)
- Další rozhraní: OLE DB, ADO.NET