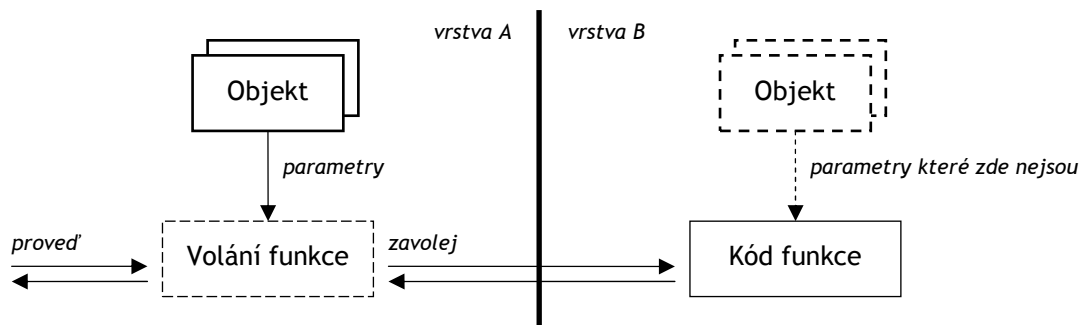


Objektově orientované distribuované prostředí

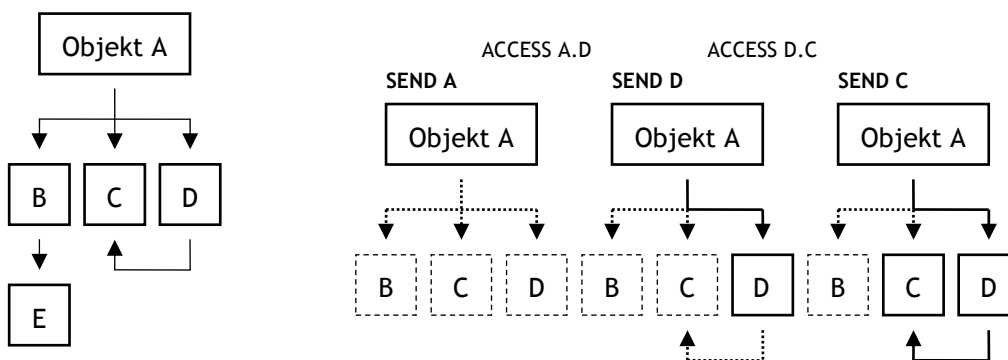
Situace

Představme si, že pracujeme na větším systému, který má více vrstev. V tomto systému už nepoužíváme jednoduché datové typy ale objektově orientovaný přístup. Zde jsou data uloženy jako objekty a protože se jedná o větší systém, tak jednotlivé vrstvy běží na jiných fyzických serverech. Teď nastává problém, protože chceme pracovat s objekty které jsou uloženy v jedné vrstvě pomocí kódu který je v druhé - většinou se jedná o vzdálené volání procedur (RPC) s objektovými parametry:



Obr.1: Jednoduchá situace

Tato situace je řešitelná - data objektu se na jedné straně převedou např. do XML a na druhé se z XML znova poskládá objekt a pak zavolá funkce. Toto jednoduché řešení má ale háček v tom, že objekt může obsahovat reference dalších objektů a přenášet všechno je někdy zbytečné, protože vzdálená funkce nemusí použít k výpočtu všechny existující data.



Obr.2: Postupné odesílání referovaných objektů

Implementace

Omezuje se na data, zpětné volání metod není možné.

Odesílací strana

- Funkce která provede prvotní volání (host:port, namespace:název+parametry, hloubka)
- Lineární seznam odeslaných objektů, které se dále již nebudou nikdy odesílat (při $h > 0$)
- Seznam nedeslaných objektů (mapování ID zástupce na objekt)
- Reference na objekty se posílají jako zástupci, s ID od 1 (viz předchozí seznam)

Přijímací strana

- Seznam všech přijatých objektů (plní účel cache, při vícenásobné referenci)
- Použití `__get` na čtení reference, tady se může požádat druhá strana o doplnění dat