



# Операційні системи

Лекція 16

Виклик віддалених процедур  
Remote Procedure Call (RPC)



# План лекції



# Концепція віддаленого виклику процедур

- Добре відомий **механізм передачі керування і даних** всередині програми, що виконується на одній машині, поширюється на передачу керування і даних через мережу
- Найбільша ефективність RPC – коли існує інтерактивний зв'язок між віддаленими компонентами з
  - невеликим часом відповідей
  - відносно малим обсягом даних, що передають
- Характерні риси локального виклику процедур
  - **Асиметричність** – одна із сторін є ініціатором взаємодії
  - **Синхронність** – процедуру, яка викликає, блокують до повернення з процедури, яку викликали
- Ідея полягає в тому, щоби віддалений виклик процедур виглядав для прикладної програми точно так, як виклик локальної процедури



# Можливі проблеми реалізації RPC

- Процедури виконуються на різних машинах, вони мають різні адресні простори, і не мають спільної пам'яті
  - Параметри не повинні містити покажчиків на пам'ять (в тому числі на стек)
  - Значення параметрів виклику слід передавати з одного комп'ютера на інший
- RPC обов'язково використовує систему обміну повідомленнями
  - Необхідно забезпечити прозорість RPC для прикладних програм
- Існує можливість аварійного завершення одного з процесів без повідомлення про це іншого, а також можливість втрати повідомлень у мережі
  - У разі краху програми, що викликає, віддалені процедури стають "сиротами"
  - У разі краху віддаленої процедури, програма, яка їх викликала, стає "обездоленою"
- Існують розбіжності у форматах подання чисел у різних архітектурах, у порядку параметрів викликів, у порядку байтів, у кодуваннях символів



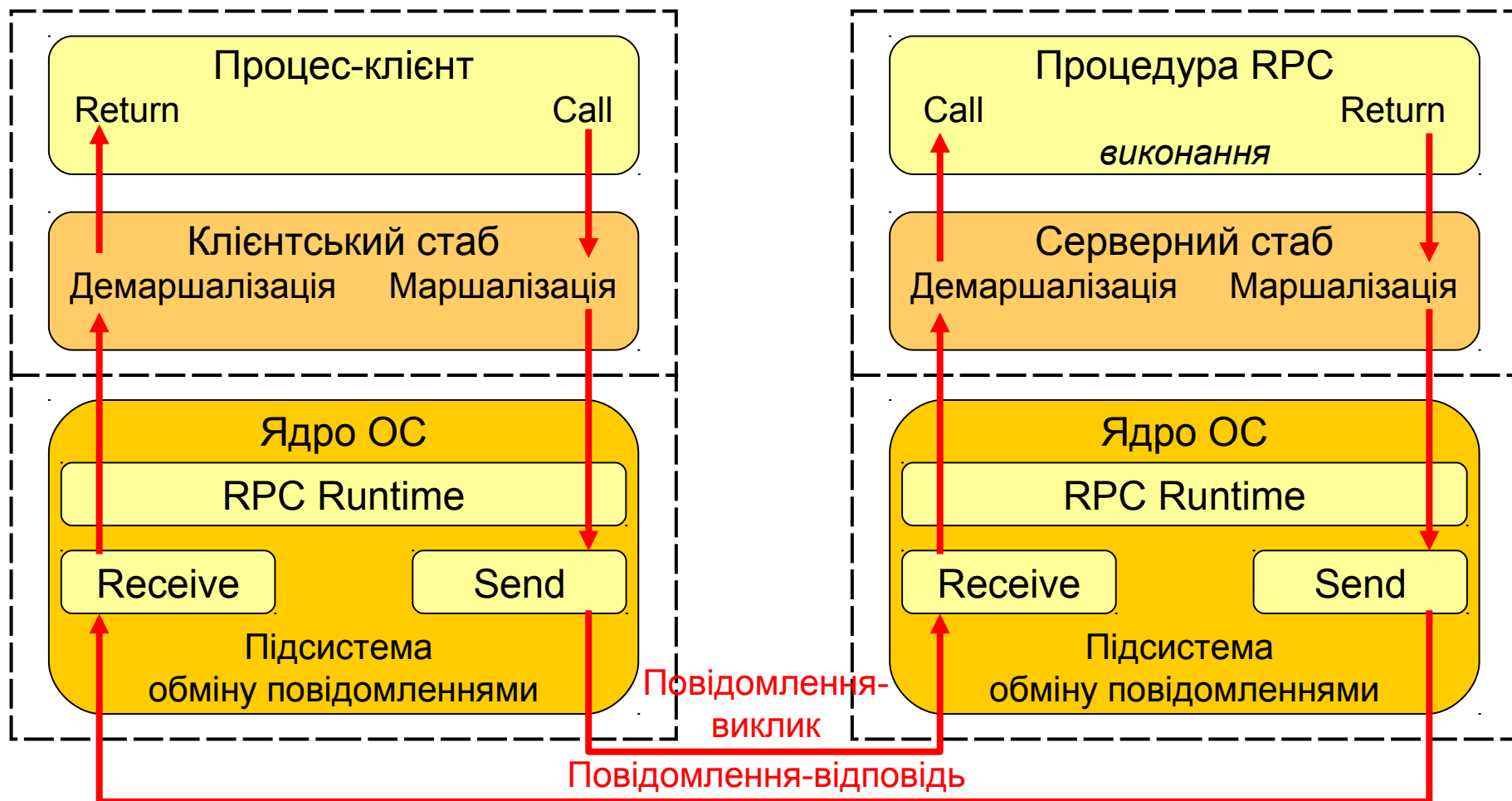
# Досягнення прозорості RPC

- В бібліотеку процедур на клієнтському комп'ютері замість коду процедури поміщають так званий **стаб** (**stub** – заглушка)
  - Клієнтський стаб викликають шляхом звичайної передачі параметрів через стек
- На комп'ютер-сервер поміщають оригінальний код процедури, а також **серверний стаб**
- Призначення клієнтського та серверного стабів – організувати передачу параметрів виклику процедури і повернення результату через мережу
  - Клієнтський стаб формує повідомлення, що містить ім'я процедури і параметри виклику (упаковка, або **маршалізація** повідомлення)
  - Серверний стаб отримує повідомлення, розпаковує (**демаршалізує**) параметри, і здійснює звичайний локальний виклик процедури
- Стаби використовують системні засоби обміну повідомленнями (send і receive)
  - Іноді у підсистемі обміну повідомленнями виділяють окремий програмний модуль для організації зв'язку стабів з примітивами обміну повідомленнями (RPC Runtime)

# Виконання віддаленого виклику процедури

Комп'ютер-клієнт

Комп'ютер-сервер





# Генерація стабів

- Ручна генерація стабів
  - Програміст використовує ряд допоміжних функцій, наданих розробниками засобів RPC
  - Є свобода вибору способу передачі параметрів виклику і застосування тих чи інших примітивів передачі повідомлень
  - Вимагає значного обсягу ручної праці
- Автоматична генерація стабів
  - Застосовується мова визначення інтерфейсу (Interface Definition Language, IDL)
  - Опис інтерфейсу між клієнтом і сервером RPC містить список імен процедур, а також список типів аргументів і результатів цих процедур
  - Опис достатній для перевірки стабом типів аргументів і генерації повідомлення-виклику
  - IDL-компілятор створює вихідні модулі клієнтських і серверних стабів, а також генерує файли-заголовки з описом типів процедур і їхніх аргументів



# Формат повідомлень RPC

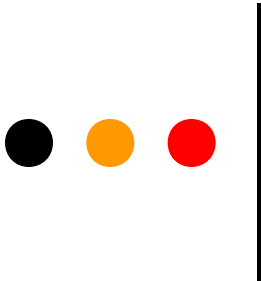
## Повідомлення-виклик

| Ідентифікатор повідомлення | Тип повідомлення | Ідентифікатор клієнта | Ідентифікатор віддаленої процедури |              |                 | Аргументи |
|----------------------------|------------------|-----------------------|------------------------------------|--------------|-----------------|-----------|
|                            |                  |                       | Номер програми                     | Номер версії | Номер процедури |           |

## Повідомлення-відповідь

| Ідентифікатор повідомлення | Тип повідомлення | Статус відповіді<br>/<br>помилка | Результат<br>або причина помилки |
|----------------------------|------------------|----------------------------------|----------------------------------|
|----------------------------|------------------|----------------------------------|----------------------------------|





# Зв'язування клієнта з сервером

- Процедуру, що встановлює відповідність між клієнтом і сервером RPC, називають **зв'язуванням** (binding)
- У різних реалізаціях RPC, можуть відрізнатись:
  - Способи завдання сервера, з яким хотів би бути зв'язаним клієнт
  - Способи визначення мережної адреси (місцезнаходження) потрібного сервера
  - Стадії, на якій відбувається зв'язування
- Статичне зв'язування:
  - Ім'я або адреса RPC-сервера задається явно
  - Відсутня гнучкість і прозорість
- Динамічне зв'язування:
  - Ім'я інтерфейсу RPC задається у вигляді <тип інтерфейсу><екземпляр інтерфейсу>
  - Тип визначає усі характеристики, крім місцезнаходження
  - Якщо клієнту важливий лише тип інтерфейсу, то процес знаходження сервера може застосовувати або **широкомовні запити**, або централізованого **агента зв'язування**