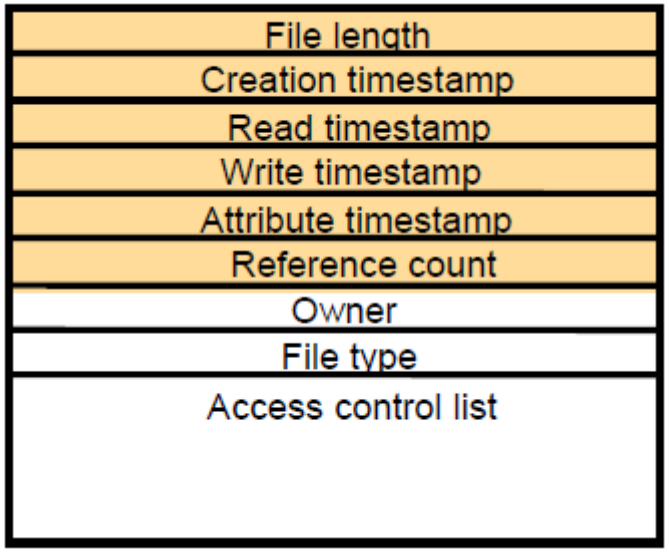
Разпределени файлови системи

Файлова система

- Файл – логическа последователност от байтове.
- Файлова система – компонент на ОС, осигуряващ поддържането на информацията на външен носител и достъпът до нея.
- Поддържа информация за файловете – File Control Block (Metadata).



Разпределена файлова система (DFS)

- Дава възможност програмите да съхраняват и достъпват отдалечени файлове по същи начин, както и локални такива.

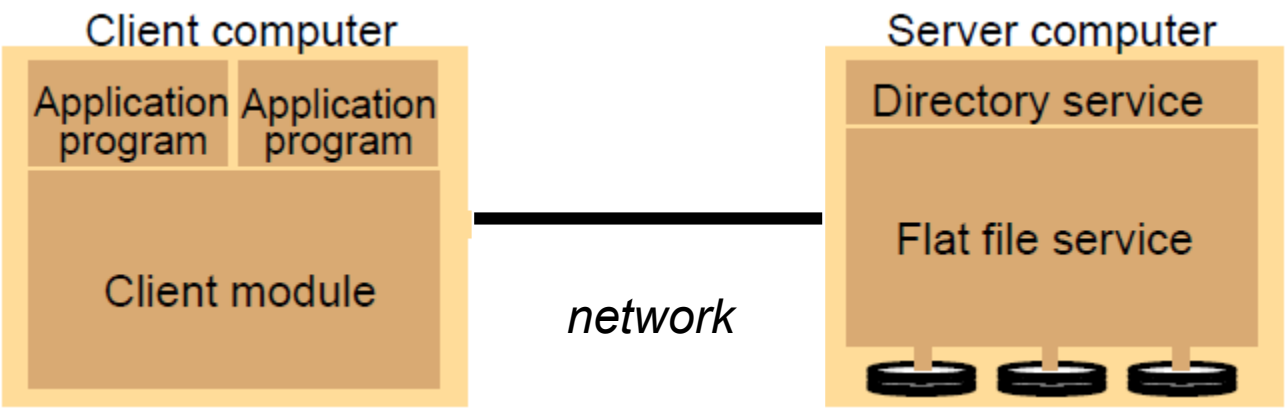
- Ресурсите на отделна машина са локални за нея. Ресурсите на другите машини са отдалечени.

- Файловата система предоставя услуги на клиентите чрез файлови операции.

Особености

- Споделяне на данни между множество потребители.
- Мобилност на потребителите.
- Прозрачност на местоположението.
- Независимост на местоположението.
- Прозрачност на разширяването.
- Репликация на файлове.

Архитектура на DFS



Flat file service

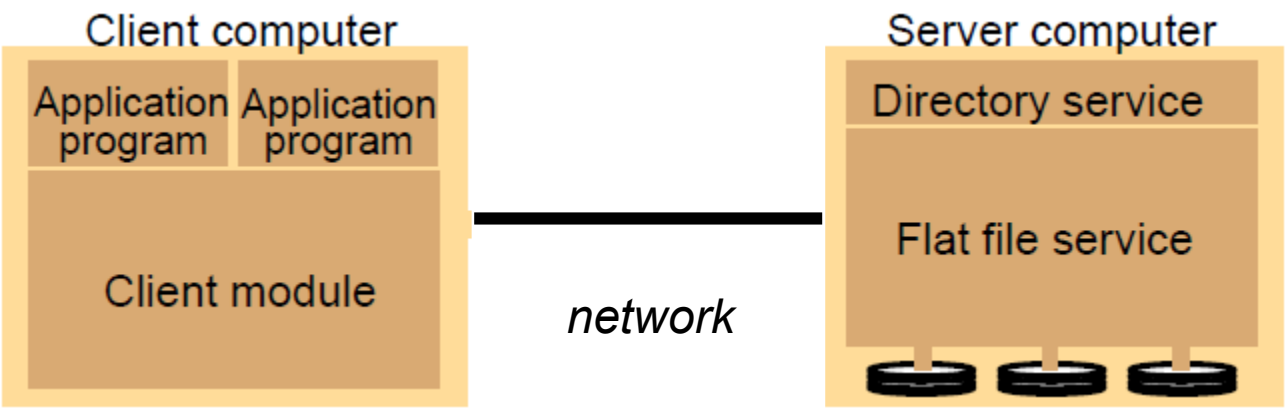
•Извършва файловите операции

•Използва Unique File ID (UFID) за указване на файлове

•Интерфейс

- RPC механизъм за извършване на операциите
- Не се използва нормално от приложенията

Архитектура на DFS (2)



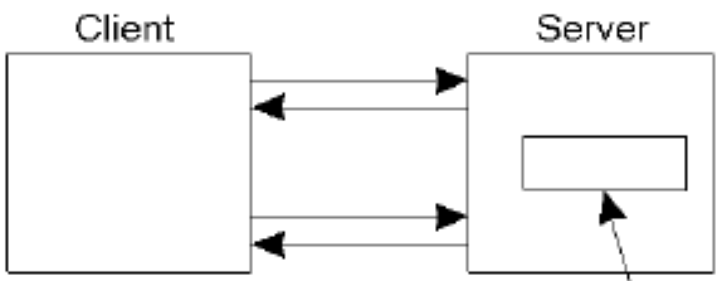
Directory service

•Свързва UFID към текстови имена на файлове и обратно

Client module

•Предоставя API за файловите операции на приложенията

Алтернативи на DFS

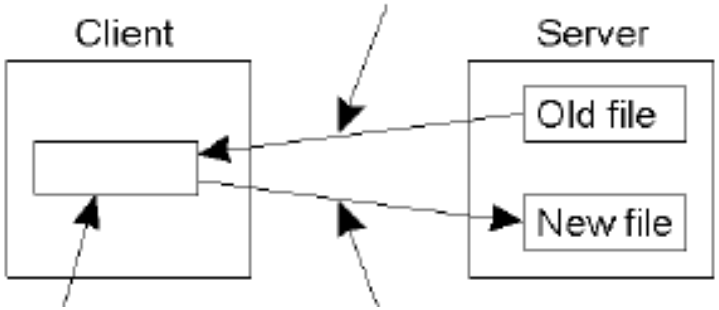


Заявки за достъп до отдалечен файл

Файлът е на сървъра

Remote access model

1. Файлът се премества при клиента



2. Достъпът се извършва от клиента

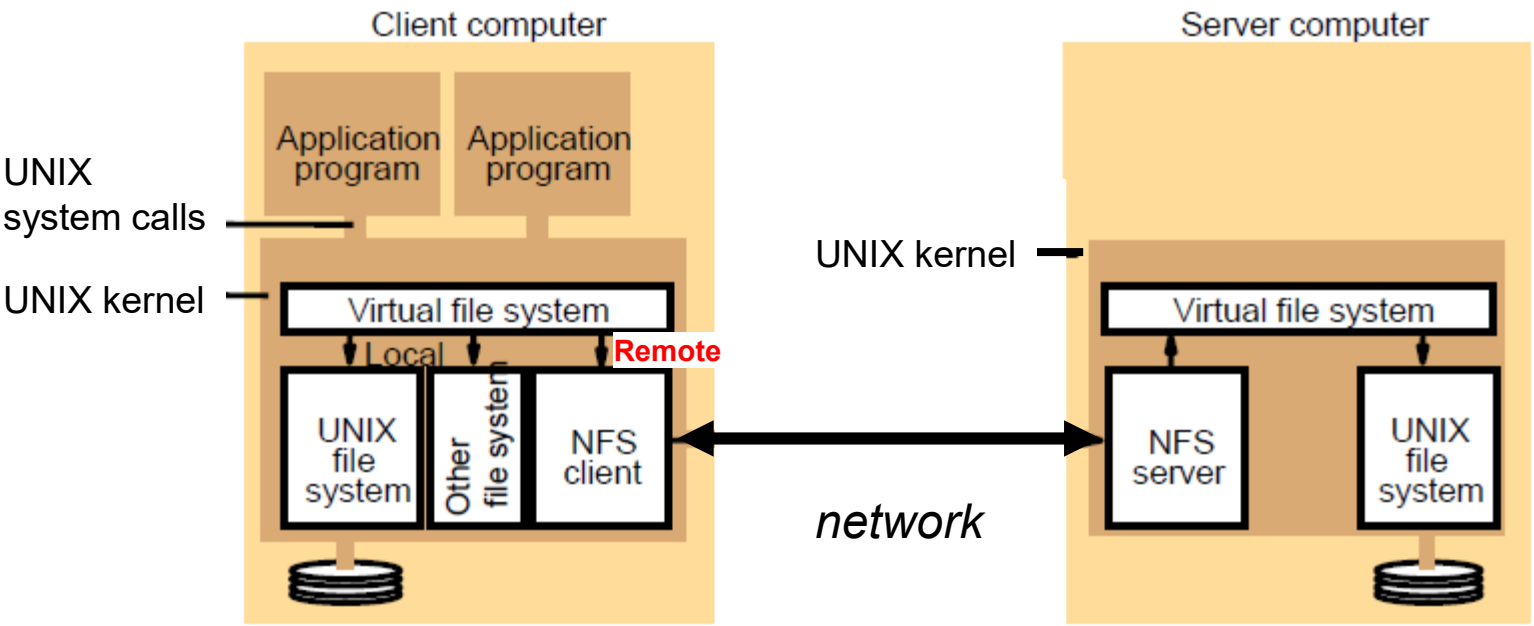
3. След действията на клиента, файлът се връща на сървъра

Download/upload model

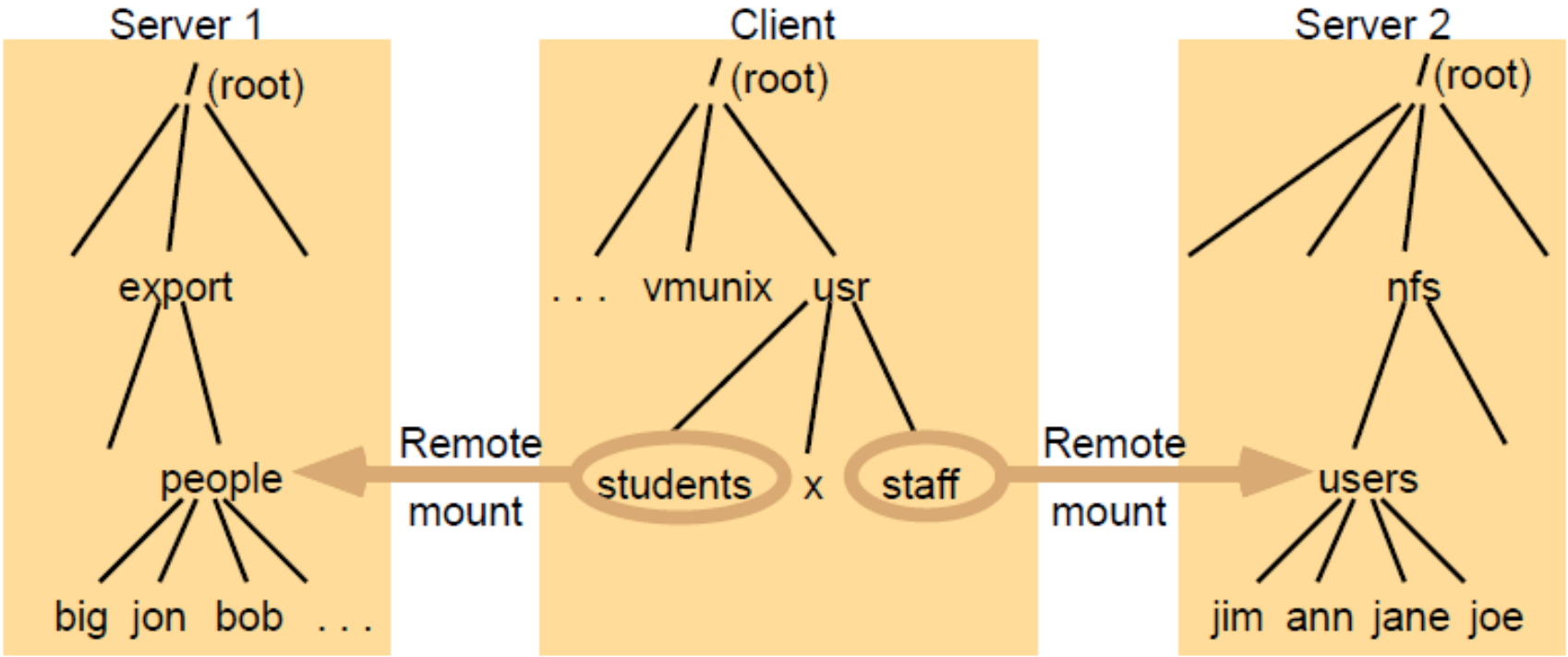
Network File System (NFS)

- Разработена от Sun Microsystems за UNIX системи.
- Stateless. Всички клиенти трябва да съхраняват информация за трансферите.
- Използва Virtual File System (VFS).
- Базира се на механизма на RPC.
- Отдалечената ФС се монтира при клиента.

Network File System (NFS)



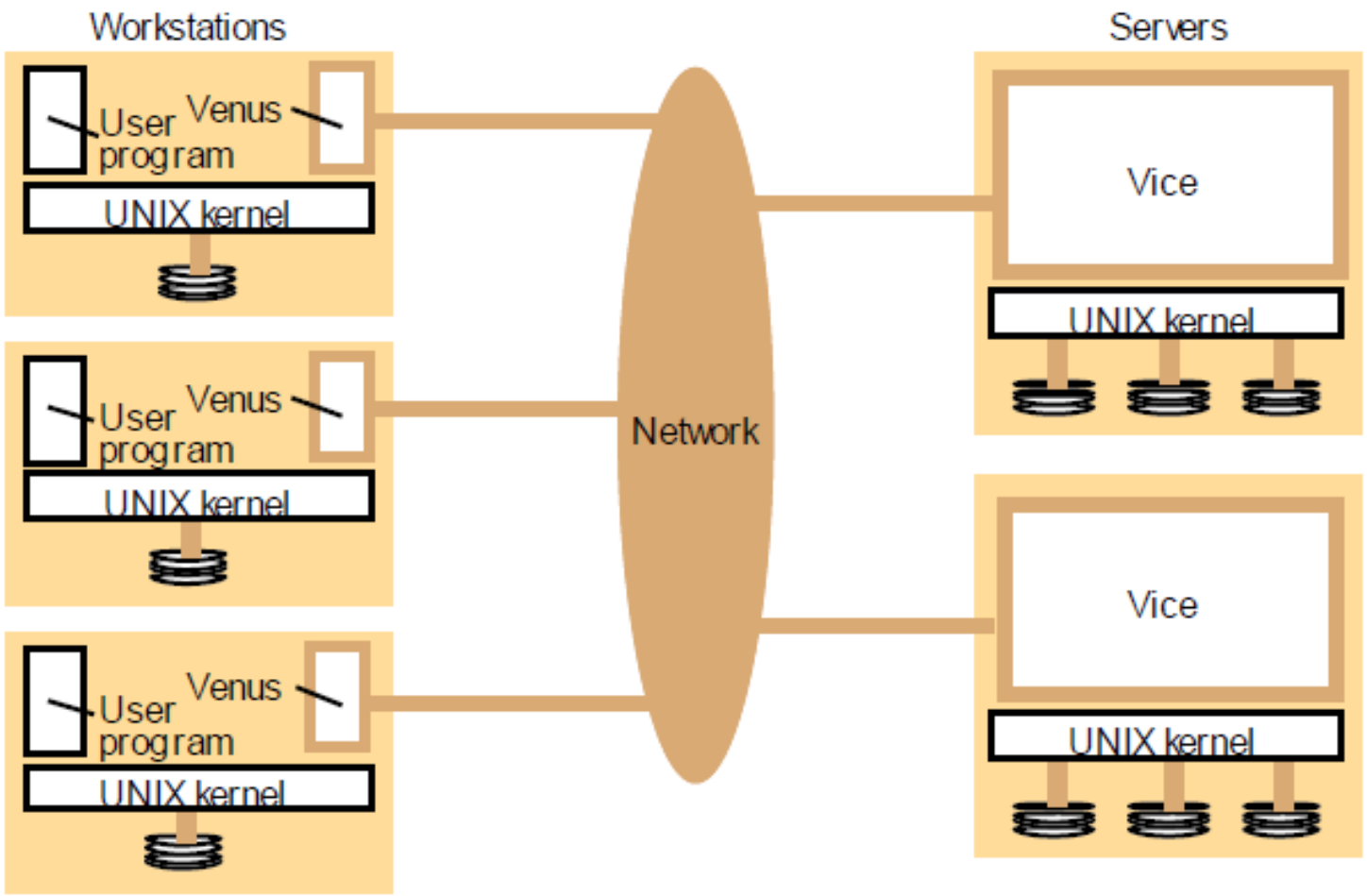
Монтиране на ФС



Andrew File System (AFS)

- Проект на Carnegie Mellon University и IBM.
- DFS, използваща множество от доверени сървъри.
- Минимизиране на комуникацията клиент-сървър.
- Прехвърляне на цели файлове между сървъра и клиент и кеширането им при клиента, докато сървърът не получи по-нови техни версии.

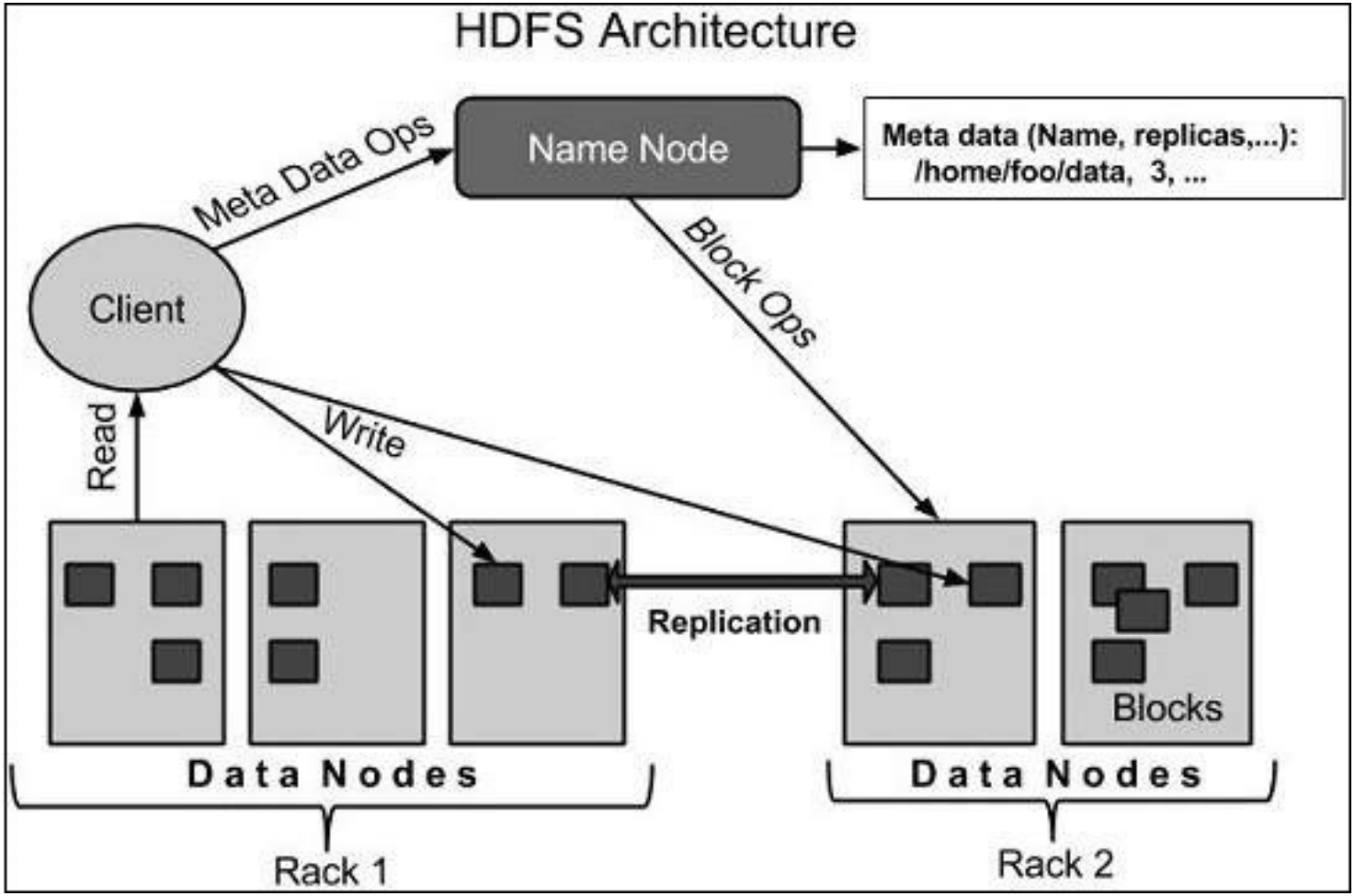
Архитектура на AFS



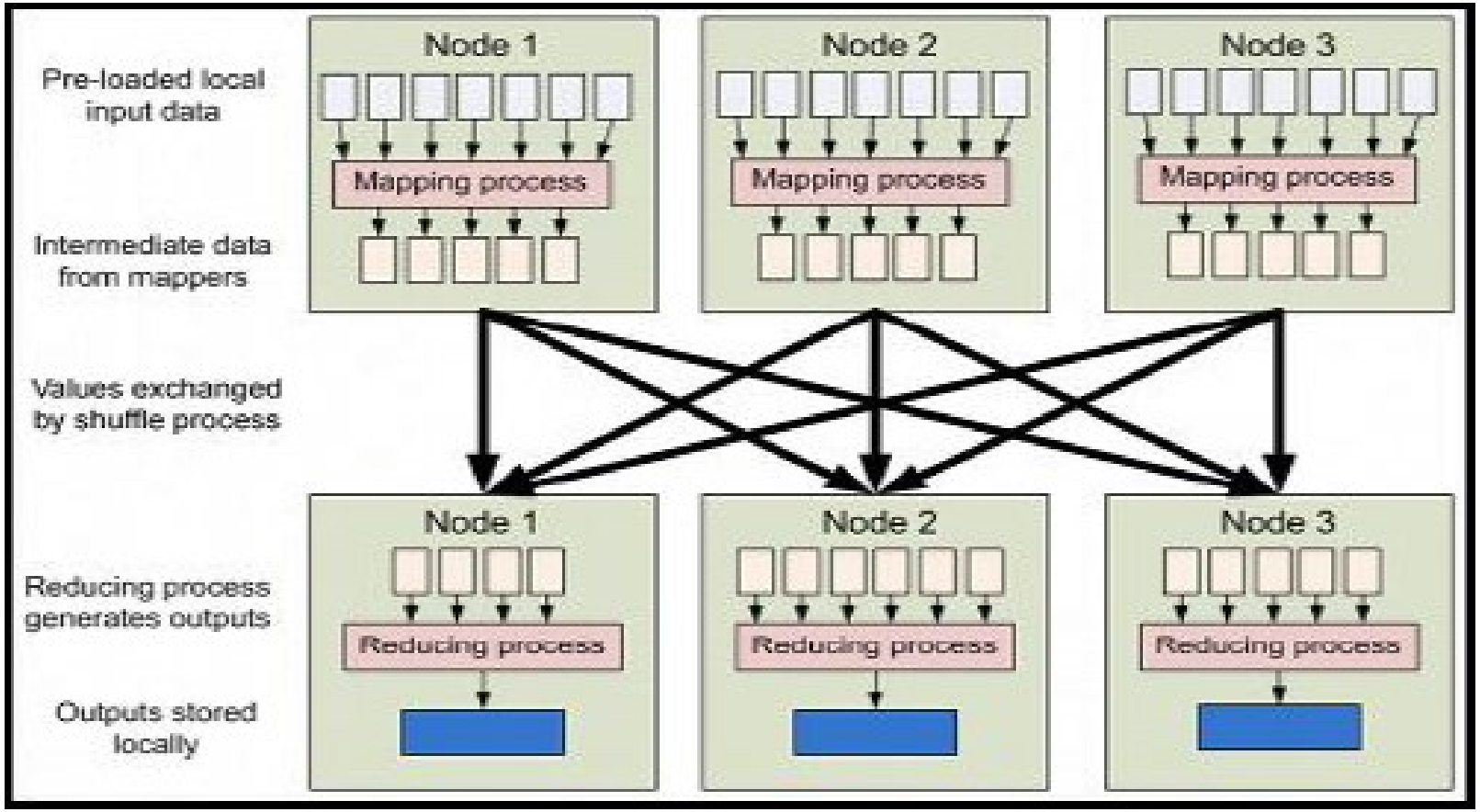
Hadoop DFS

- Управлява съхранението на данните в Hadoop клъстер.
- Проектирана за надеждно съхранение на големи обеми данни (BigData).
- Файловете се разделят в блокове (един или няколко), които се съхраняват в даннови възли. Размерът на блок е 64MB по подразбиране.
- Предоставя висока bandwidth на потребителски приложения.
- Дава възможност за паралелно изпълнение на приложенията.
- Изпълнява се върху широко разпространени компютри.

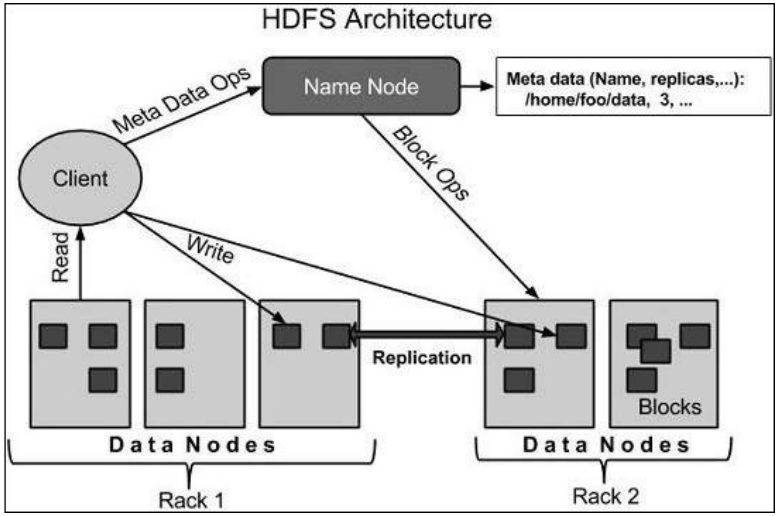
Архитектура на HDFS



Hadoop клъстер

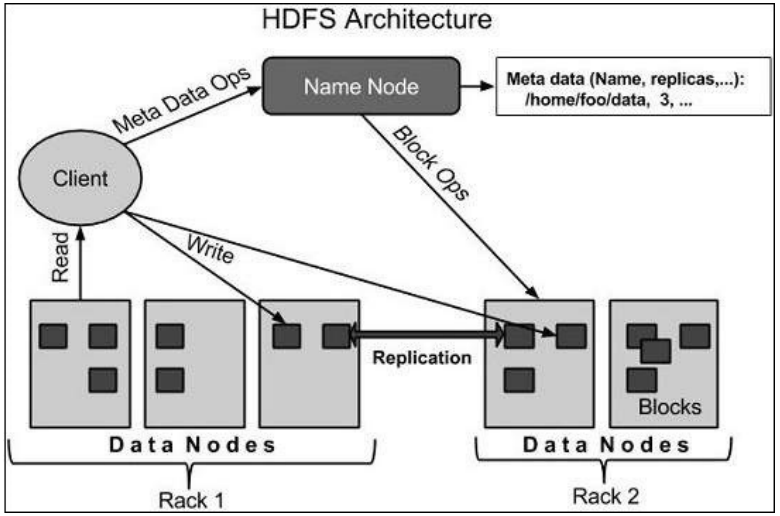


Архитектура на HDFS (2)



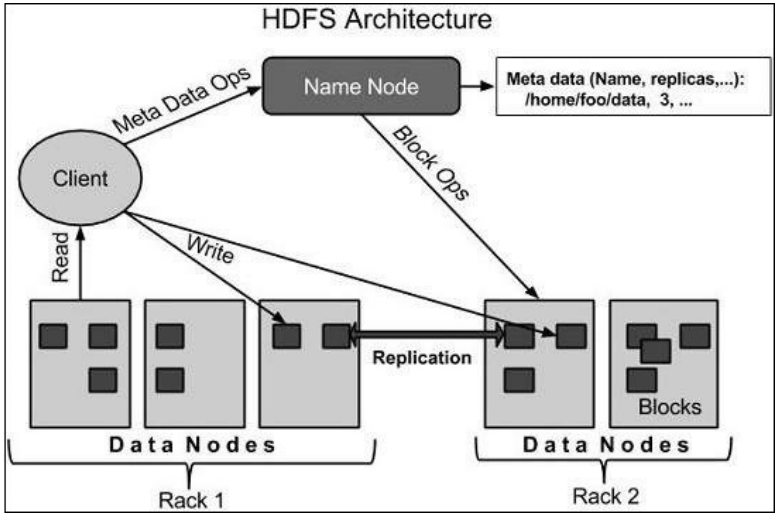
- Metadata – метаданни (атрибути като разрешения за достъп, модификации, времена на достъп, дискови квоти и др.) и списък с блокове за файл.
- Метаданните се съхраняват на отделен сървър – NameNode (Master).
- Приложните данни се съхраняват на други сървъри – DataNodes (Slave).
- Всички сървъри са напълно свързани и комуникацията е върху TCP. Използва се RPC.

NameNode



- Специален софтуер, изпълняващ се под Linux OS.
- Функционира като Master сървър:
  - Управлява пространството на имената на ФС.
  - Контролира достъпа на клиентите до файловете
  - Изпълнява системни операции като преименуване, отварян затваряне на файлове и директории.

DataNode

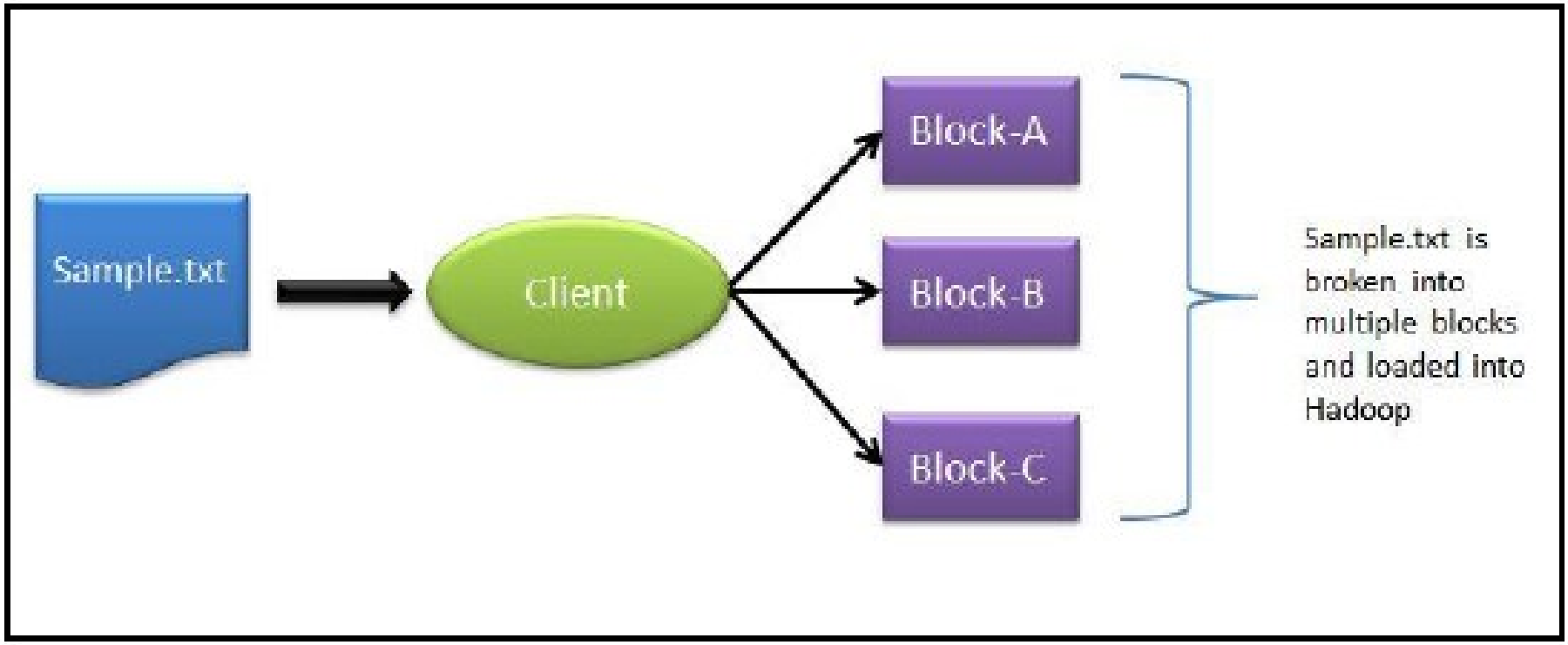


- Специален софтуер, изпълняващ се под Linux OS.
- За всеки възел в клъстера има отделен DataNode възел.
- Управлява съхранението на данните в локалния възел:
    - Изпълнява read-write операции във ФС по заявка на клиент.
    - Изпълнява системни операции като създаване, изтриване и репликации на базата на инструкции от NameNode.
    - Периодично изпращат block report към NameNode.

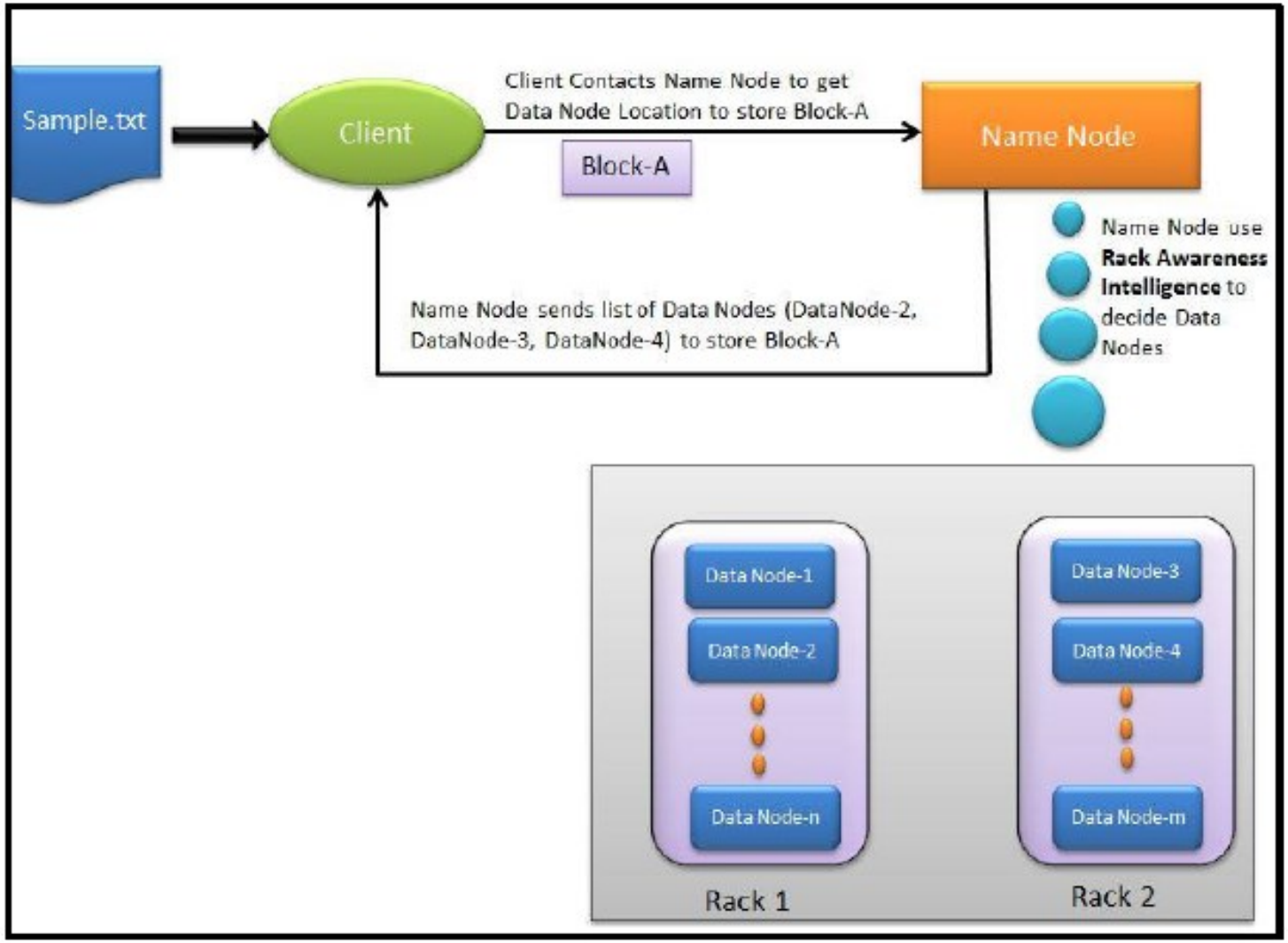
Функциониране на HDFS

- Как се зарежда файл в Hadoop клъстер?
- Как клиент разбира в кои DataNodes се зареждат блоковете?
- Кой извършва репликацията на блоковете?

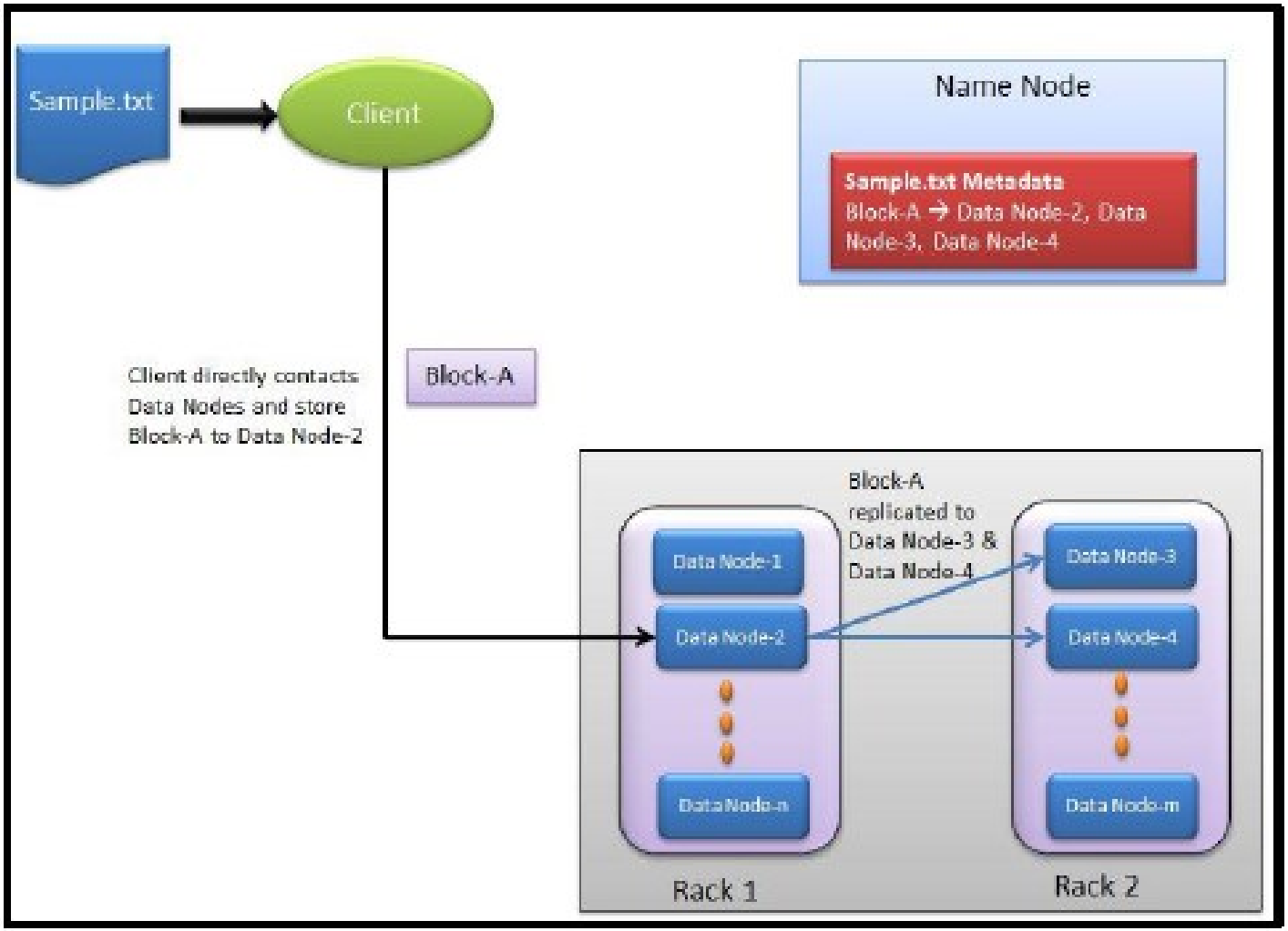
Зареждане на файл



Определяне на DataNode



Реплициране на блокове



Google File System (GFS)

- Огромни обеми данни Pb.
- Множество разпределени сървъри.
- Хиляди заявки за секунда.
- Копия на документи.

➢ Необходимост от: голяма, разпределена и отказоустойчива файлова система

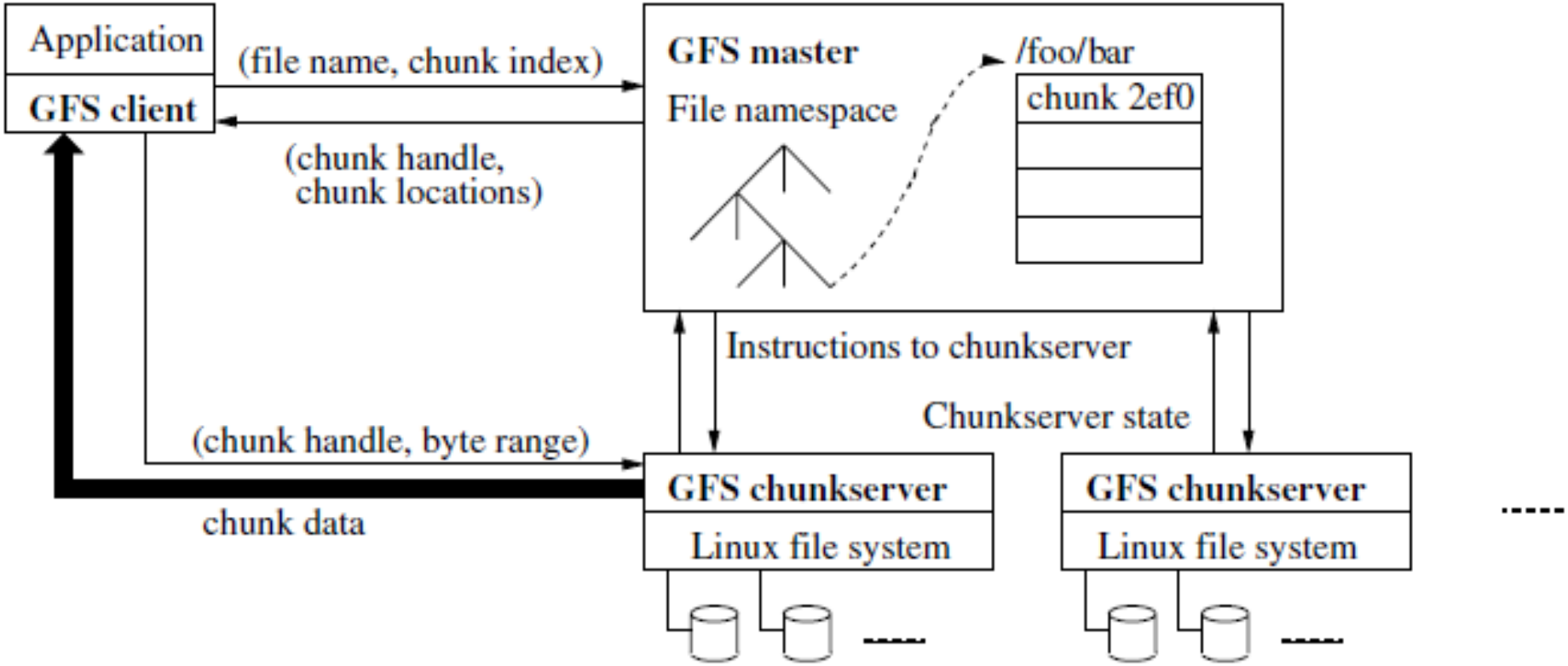
Google File System (GFS)

- Отказоустойчивостта и автоматичното възстановяване трябва да са вградени.
- Нов размер на блок.
- Добавянето на записи е по-разпространено от чист запис (Record Append).
- Приложенията на Google и файловата система трябва да бъдат съвместно проектирани.

Google File System (аналогия)

- При ФС на една машина:
  - Горните нива поддържат метаданни
  - Долното ниво (диск) съхранява данните в единици – *блокове*
- При GFS:
  - Master процес поддържа метаданни
  - Долното ниво (множество от *chunkservers*) съхранява данните в единици - *chunks*

Google File System - архитектура



Chunk

- Голям блок - 64Mb.
- Съхранява се от chunkserver като файл.
- Използват се имена за референциране (chunk handle).
- Всеки chunk е репликиран между няколко chunkservers.

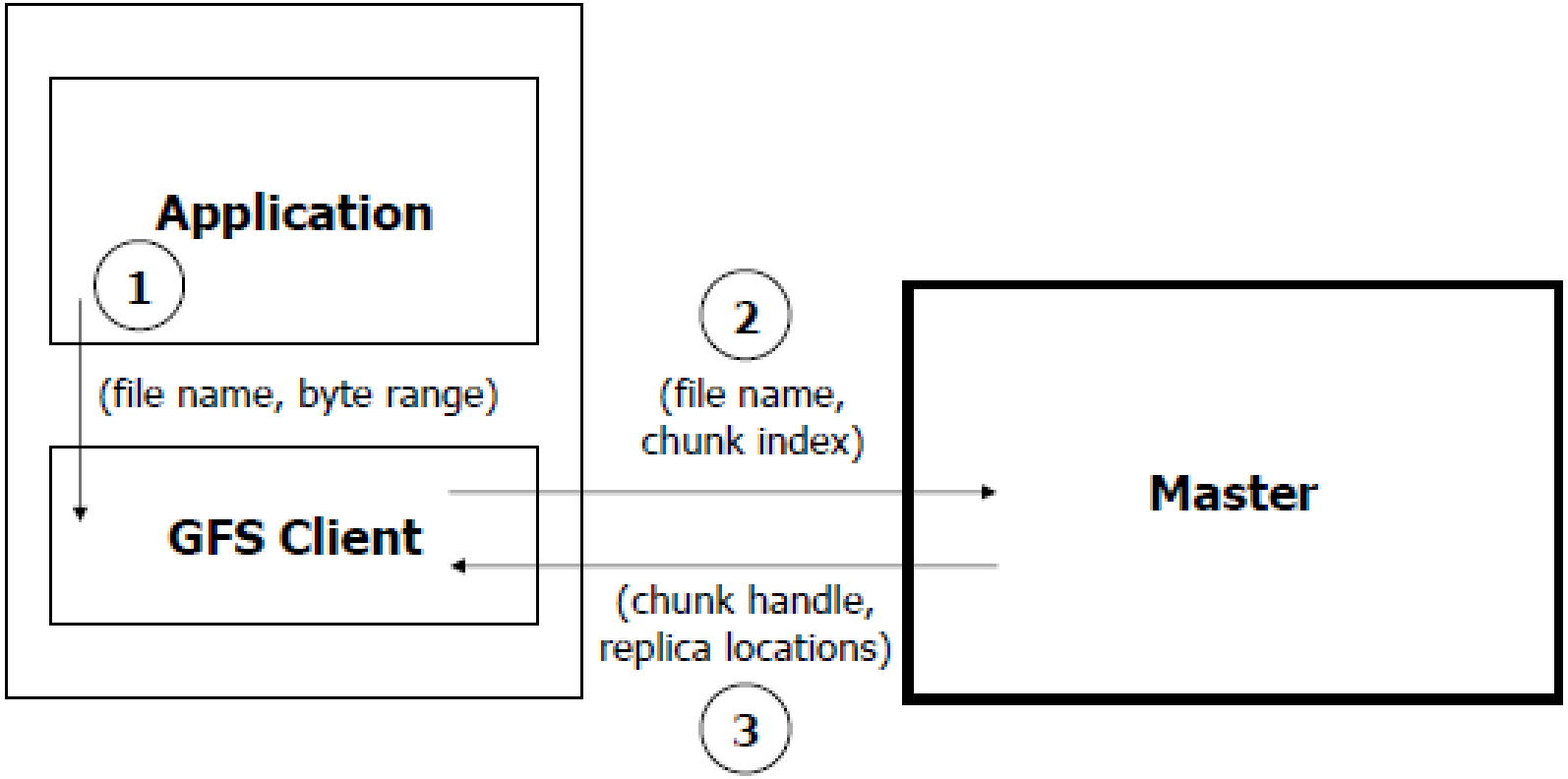
Master

- Единичен процес, изпълняващ се на отделна машина, съхраняващ *метаданни*:
  - Имената на файловете
  - Съответствията chunk - файл
  - Информация за местоположението на chunk
  - Информация за контрол на достъпа
  - Номер на версия на chunk

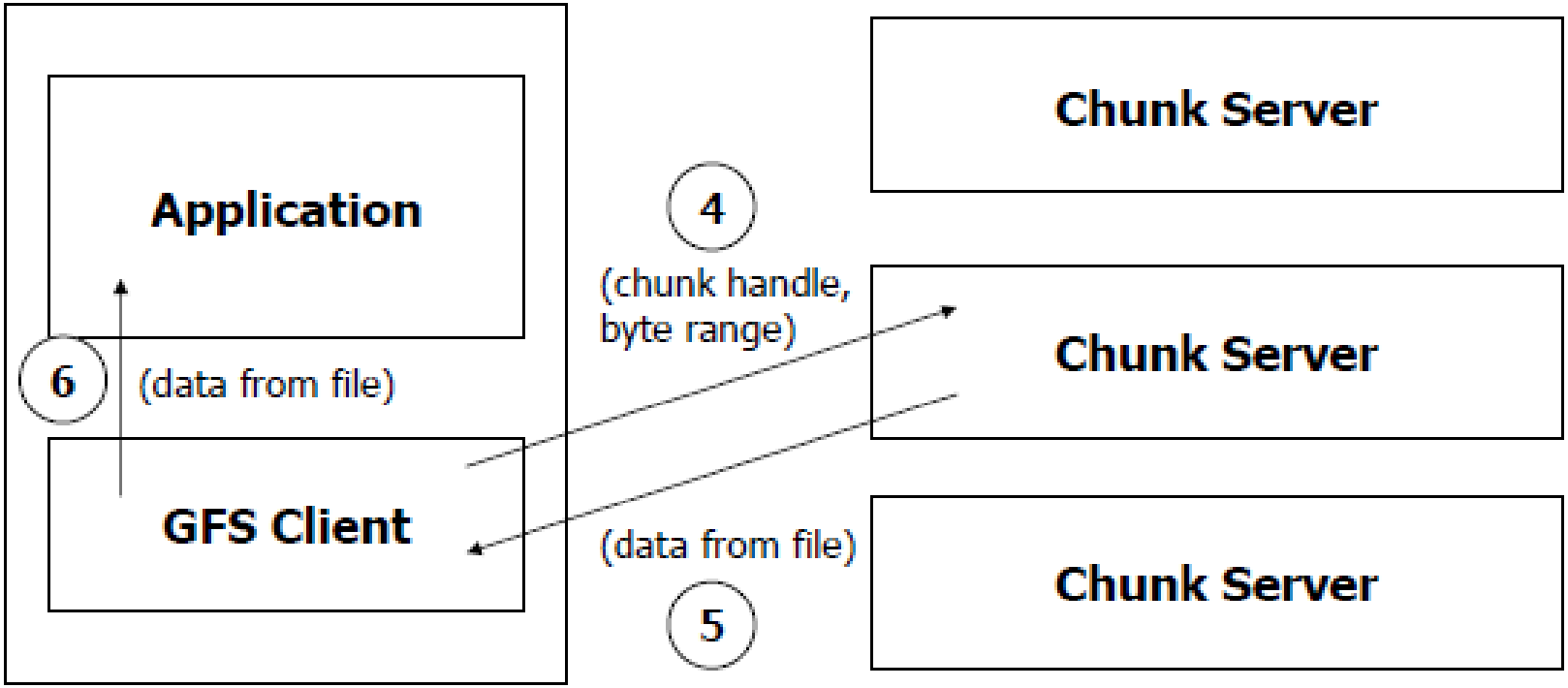
Комуникация между Master и ChunkServer

- Комуникират си регулярно (мониторинг) за получаване състоянието на chunk server:
  - Отпаднал ли е сървъра
  - Има ли отказ на диск на сървъра
  - Има ли повредени реплики
  - Кои реплики съхранява сървъра
- Master изпраща инструкции за:
  - Изтриване на съществуващ chunk
  - Създаване на нов chunk

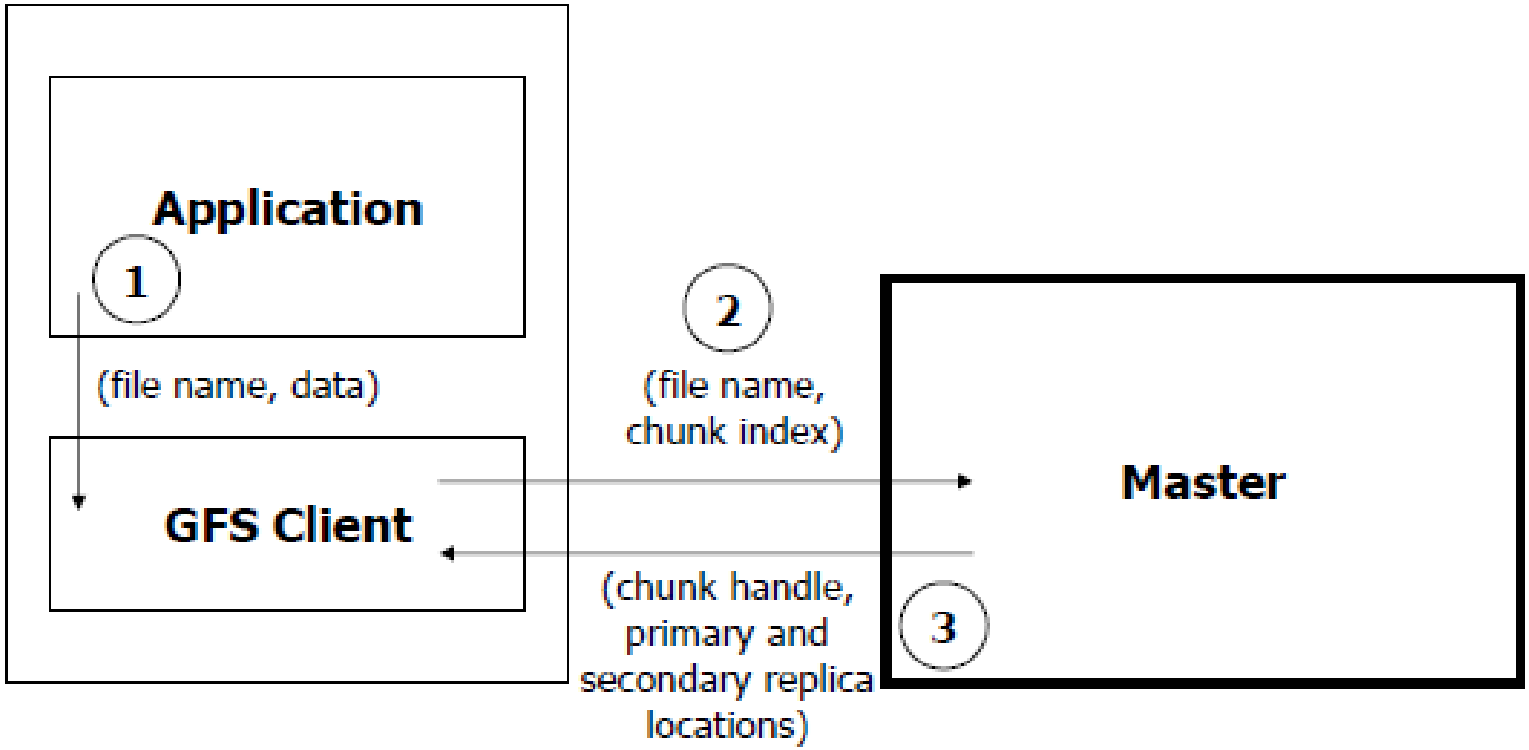
Обслужване на заявки - Read



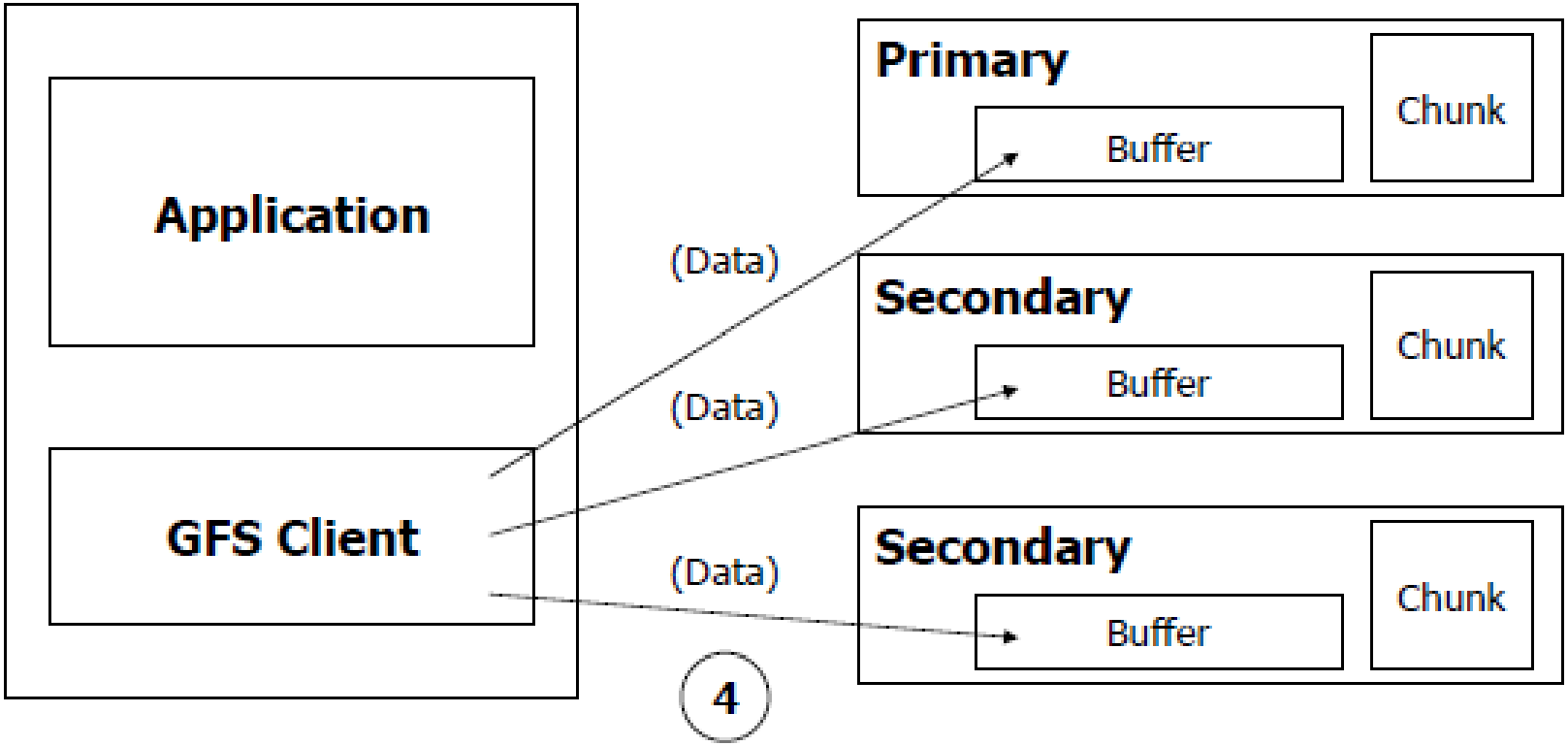
Обслужване на заявки - Read (2)



Обслужване на заявки – Write

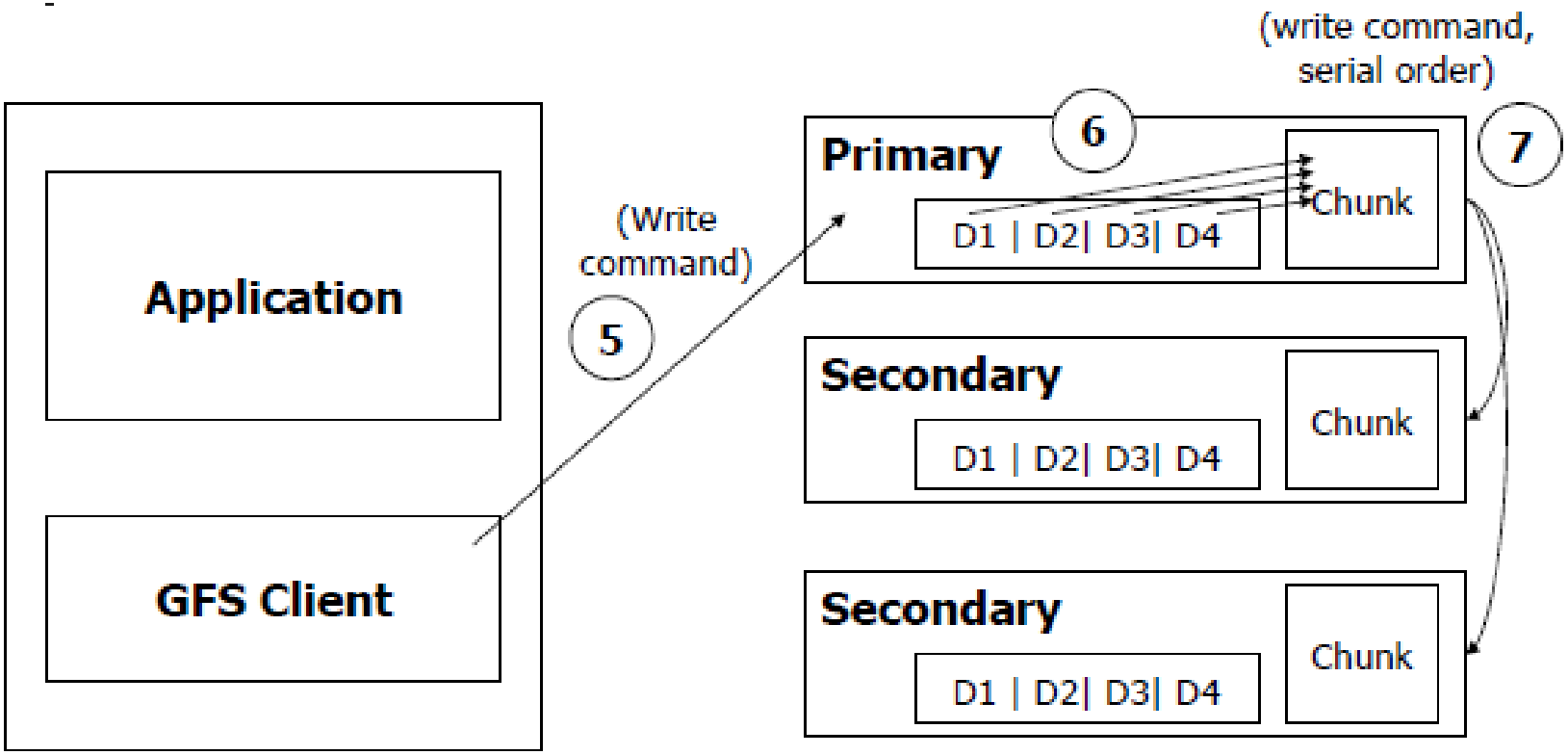


Обслужване на заявки – Write (2)

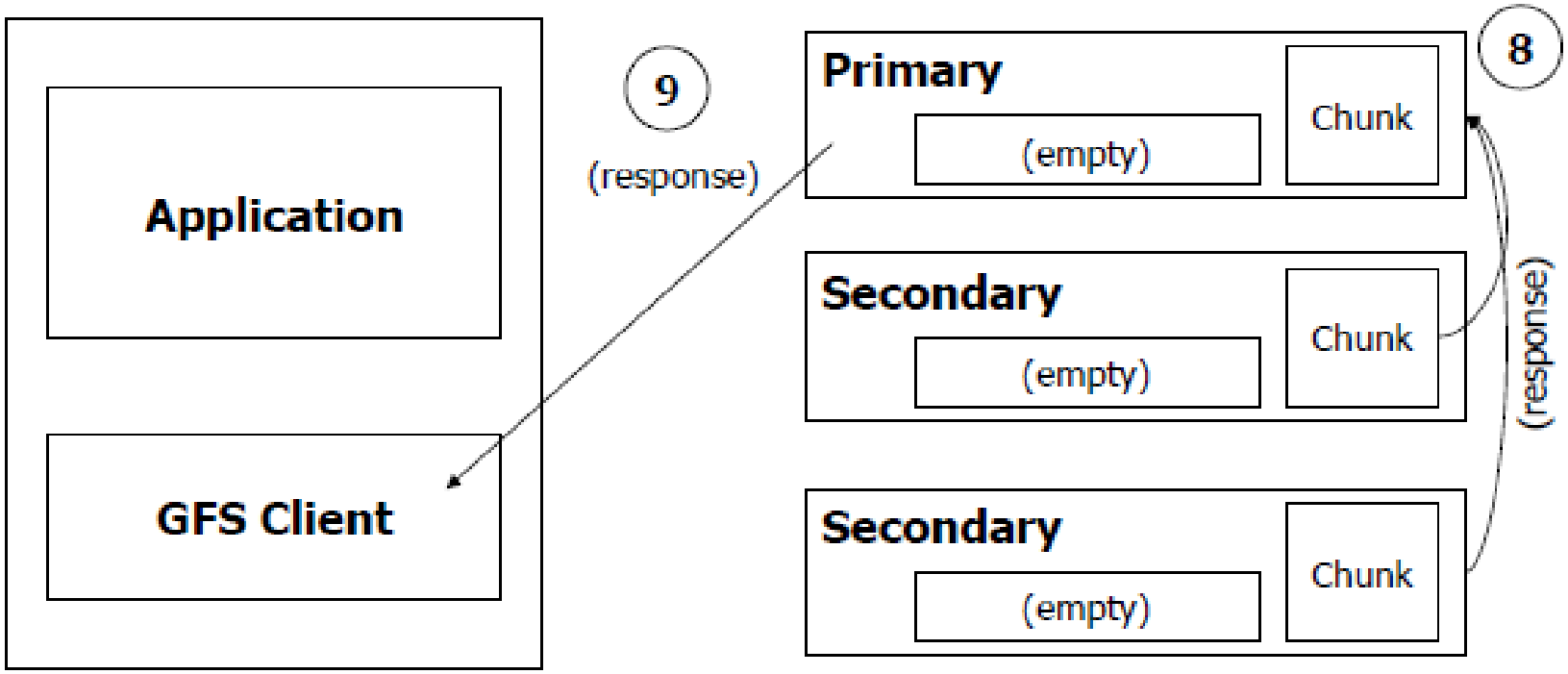


Обслужване на заявки – Write (3)

-



Обслужване на заявки – Write (4)



Record Append

Специфична операция за Google:

- Обединява резултати от множество машини в един файл.
- Използва файла като опашка от тип *производител-консуматор.*
- Клиентът изпраща данни без отместване (GFS добавя данните като последователност от байтове).
- GFS осигурява добавяне на данните атомарно.

Отказоустойчивост

- Бързо възстановяване – master и chunkservers са проектирани да се рестартират в рамките на няколко секунди.
- Репликация на chunk-овете.
- Всички промени по метаданни – в Log.
- Интегритет на данните – всеки chunk има контролна сума.

Въпроси?