Обработка и исполнение запросов в СУБД (Лекция 4)

Классические системы: введение в распределенные СУБД

v4

Георгий Чернышев

Высшая Школа Экономики

chernishev@gmail.com

23 сентября 2020 г.

Распределенные СУБД I

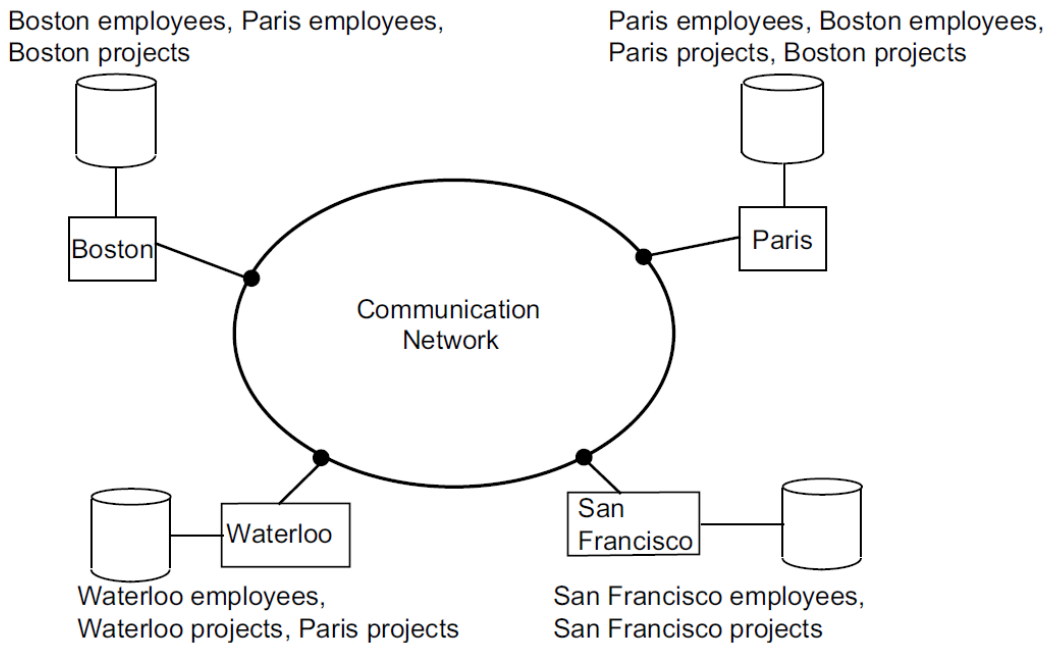


Fig. 1.5 A Distributed Application

[1]

РСУБД [Elnikety, 2009]: СУБД управляющая базой данных, данные которой подвергнуты фрагментированию и реплицированию, и находятся на нескольких, связанных друг с другом, узлах.

Основная цель РСУБД — "спрятать" физическую распределенность от клиентов. Клиенты работают с одной базой, видимой как единое целое;

[1] Изображение взято из [Özsu and Valduriez, 2009]

Распределенные СУБД II

История:

- Появились в конце 70х, вместе с shared-nothing системами;
  - Нужда больших организаций иметь несколько офисов;
  - Основная проблема в те годы: медленность сетей;
- Первые системы, примеры [Kian-Lee Tan, 2009]:
  - Исследовательские проекты: SDD-1 (Computer Corporation of America, 1980), Distributed INGRES (University of California at Berkeley, 1986), R*STAR (IBM, 1981);
  - Коммерческие продукты: INGRES/Star (1987), Oracle (1987), IBM продукты для интеграции разных версий DB2.

Что может дать РСУБД? I

Транспарентность при управлении данными.

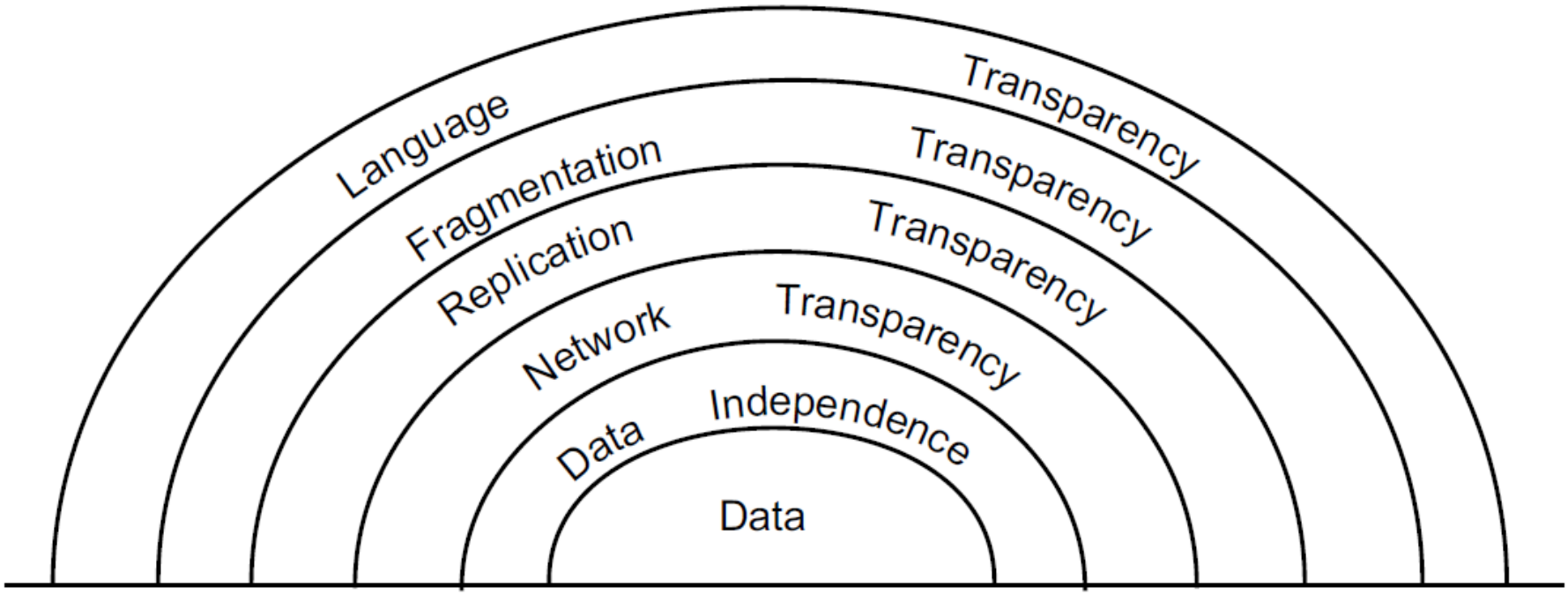


Fig. 1.6 Layers of Transparency

2

[2] Изображение взято из [Özsu and Valduriez, 2009]

Транспарентность в РСУБД

Пользователь должен считать что работает с централизованной СУБД. Для этого надо обеспечить [Özsu and Valduriez, 2009]:

- Транспарентность сети:
  - транспарентность локации;
  - транспарентность имен;
- Транспарентность фрагментирования данных;
- Транспарентность репликации данных;
- Языковую транспарентность.

В [Kian-Lee Tan, 2009] выделяется еще и транспарентность выполнения транзакций.

Что может дать РСУБД? II

Еще [Özsu and Valduriez, 2009]:

- Надежность посредством распределенных транзакций;
- Повышение производительности:
  - локализация данных: 1) обработка только части данных и 2) меньше сетевые задержки;
  - естественный параллелизм: 1) межзапросный и 2) внутризапросный;
- Расширяемость системы;

Фрагментирование данных

Бывает:

- Горизонтальное, “подклеиваем” с низу с помощью UNION;
- Вертикальное, “подклеиваем” сбоку с помощью JOIN;
- Смешанное.

Выбор оптимальной схемы фрагментирования это *NP*-трудная задача, обычно отдается на откуп администратору (ручной труд), вертикальное сложнее.

Системы лучше поддерживают горизонтальное: PostgreSQL (набор горизонтальных фрагментов), Oracle (наведенное горизонтальное фрагментирование, partition by reference) и т.д.

РСУБД, сложности

РСУБД бывают:

- Полностью реплицированные: на каждом узле есть вся база;
- Полностью фрагментированные: каждый узел содержит свой уникальный фрагмент, фрагменты попарно не пересекаются, объединение дает всю базу;
- Что-то среднее.

Основные исследовательские задачи связанные с РСУБД:

- Как хранить данные?
- Как при обновлениях синхронизировать копии?
- Как осуществлять координацию и коммуникации между узлами при исполнении запроса?

Компоненты РСУБД

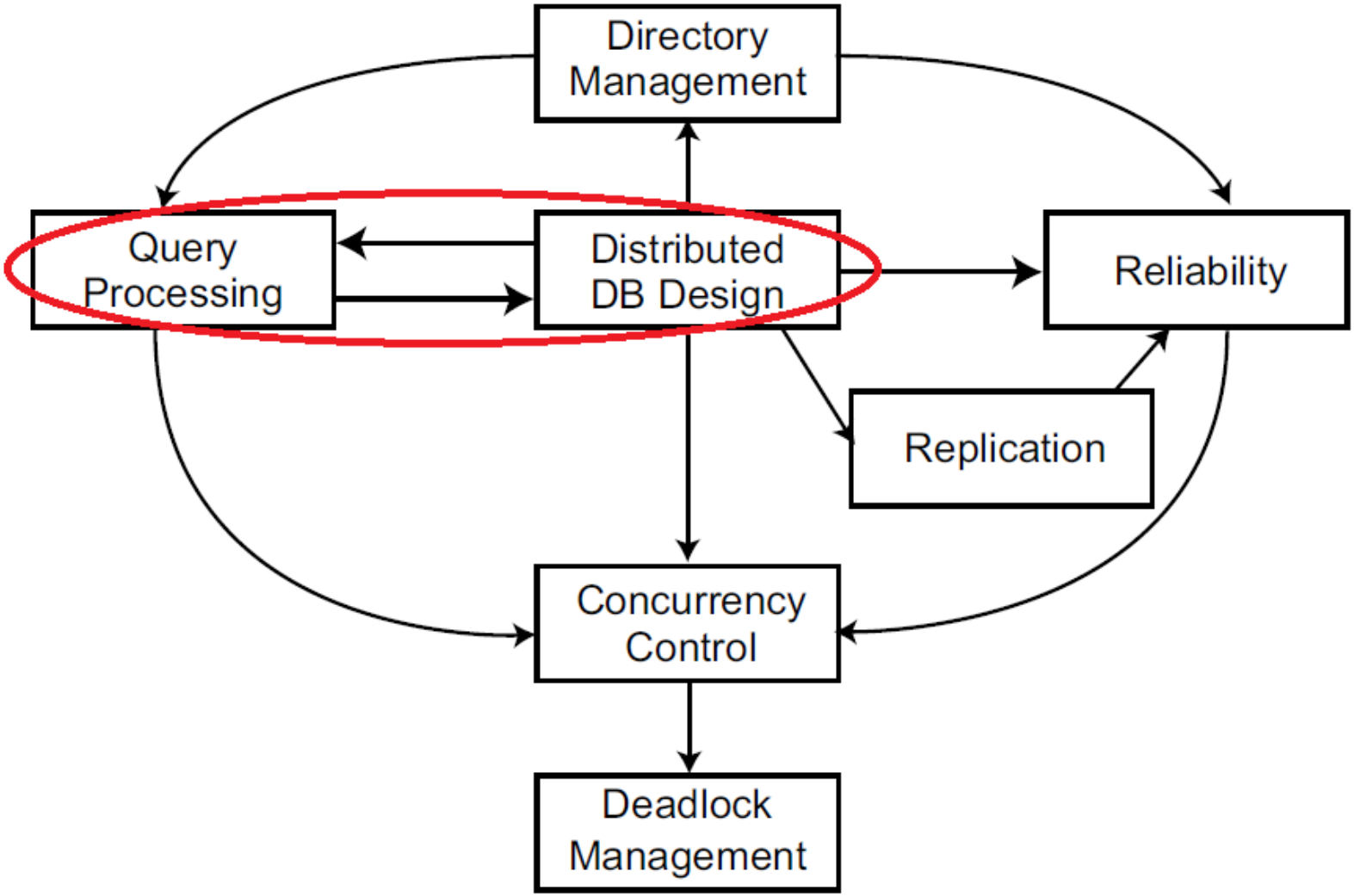


Fig. 1.7 Relationship Among Research Issues

3

[3] Изображение взято из [Özsu and Valduriez, 2009]

Какие бывают РСУБД

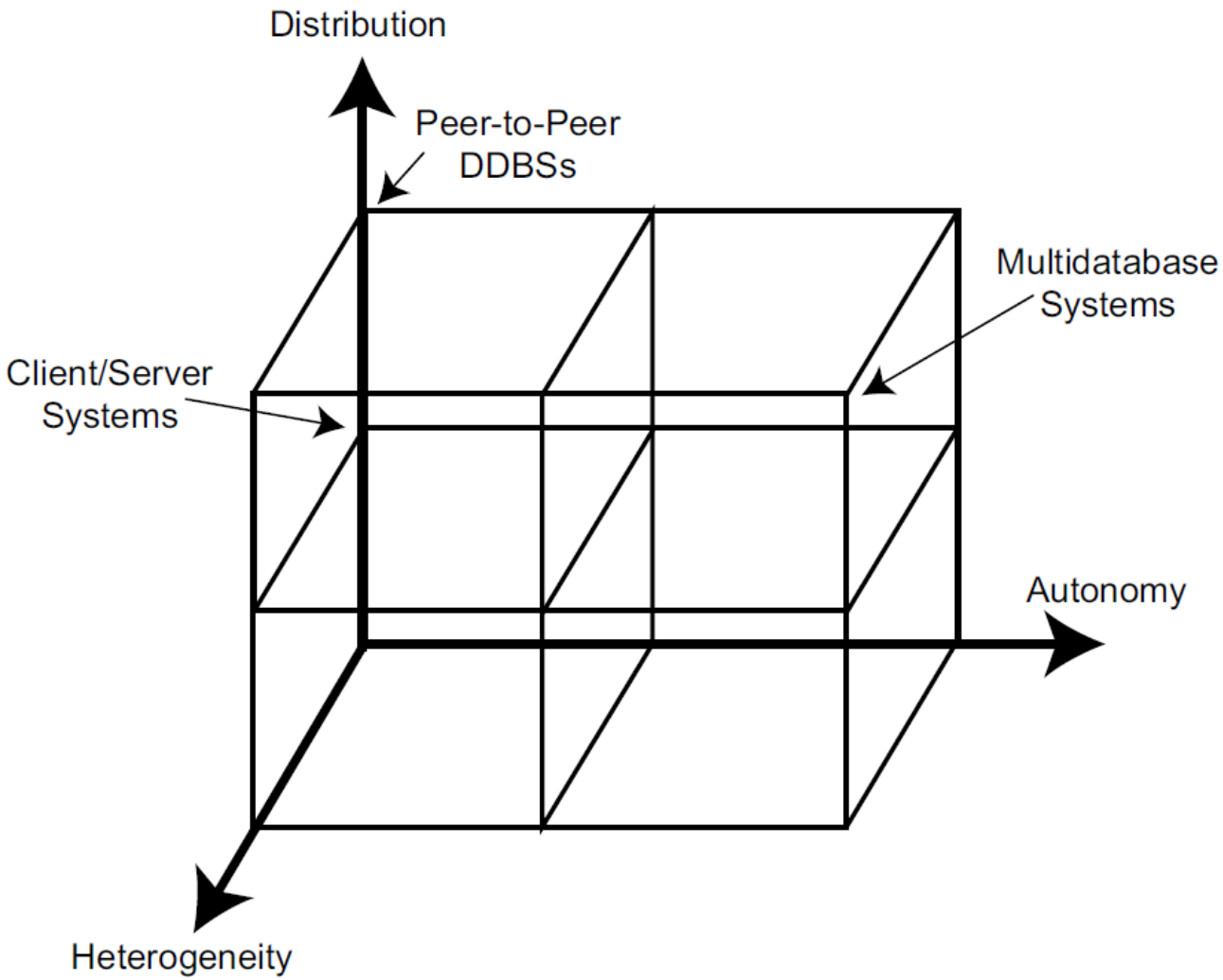


Fig. 1.10 DBMS Implementation Alternatives

[4]

[4] Изображение взято из [Özsu and Valduriez, 2009]

Аспекты РСУБД: автономность I

Автономность, в смысле автономности управления, не данных.

Показывает насколько могут отдельные СУБД работать по отдельности.

Классические требования к автономности:

- Распределенная система не влияет на локальные операции отдельных СУБД;
- Выполнении глобальных запросов не влияет на способы выполнения и оптимизации локальных запросов;
- В случае добавления или выбывания отдельных СУБД консистентность работы РСУБД не должна страдать.

Сферы приложения автономности:

- Автономность дизайна БД: СУБД-участники могут использовать любые модели данных и способы управления транзакциями;
- Автономность коммуникации: СУБД-участник решает какую информацию предоставлять другим СУБД или другому управляющему ПО;
- Автономность выполнения: СУБД-участник решает как выполнять свои транзакции.

Аспекты РСУБД: автономность II

Градации:

- Тесная интеграция — пользователю предоставляется единое видение базы, даже если она раскидана по локальным базам. Локальные базы не работают по-отдельности.
- Полуавтономная система — локальные системы могут работать независимо от наличия глобальной. Федерация баз — сами решают какие части делать доступными для пользователей других баз. Обычно требуют модификации кода.
- Полностью изолированная система — stand-alone системы, друг о друге не знают.

Это не единственный альтернативы, но самые популярные.

Аспекты РСУБД: распределенность

Распределенность именно в контексте распределенности по данным.

Типы систем:

- Клиент-серверная: управление данными на серверах, обеспечение пользовательского доступа — на клиентах;
- Peer-to-peer: нет различия между типами машин;
- Отсутствие распределенности.

Аспекты РСУБД: гетерогенность

Гетерогенность:

- Хардварная;
- Сетевая;
- Моделей данных;
- Языков запросов;
- Протоколов управления транзакциями;
- ...

В рассматриваемой классификации она либо есть, либо нет.

Основные типы РСУБД I

Типы:

- (A0,D1, H0) клиент-сервер РСУБД;
  - Клиент-серверность: не процесс, а разделение машин;
  - Появились в 90е;
  - Сервер: основная работа;
  - Клиент: Application Interface, UI, кеши данных, кешированные замки транзакций;
  - модели:
    - много клиентов - один сервер;
    - много клиентов - много серверов;
  - много клиентов - много серверов:
    - тяжелый клиент — клиент сам выбирает куда подсоединяться когда надо, это упрощает код сервера но огружает клиентские машины;
    - легкий клиент — клиент помнит о “домашнем” сервере, сервер делает всю работу по взаимодействию с другими серверами;
  - трехуровневая и многоуровневая архитектура.

Клиент-серверная СУБД

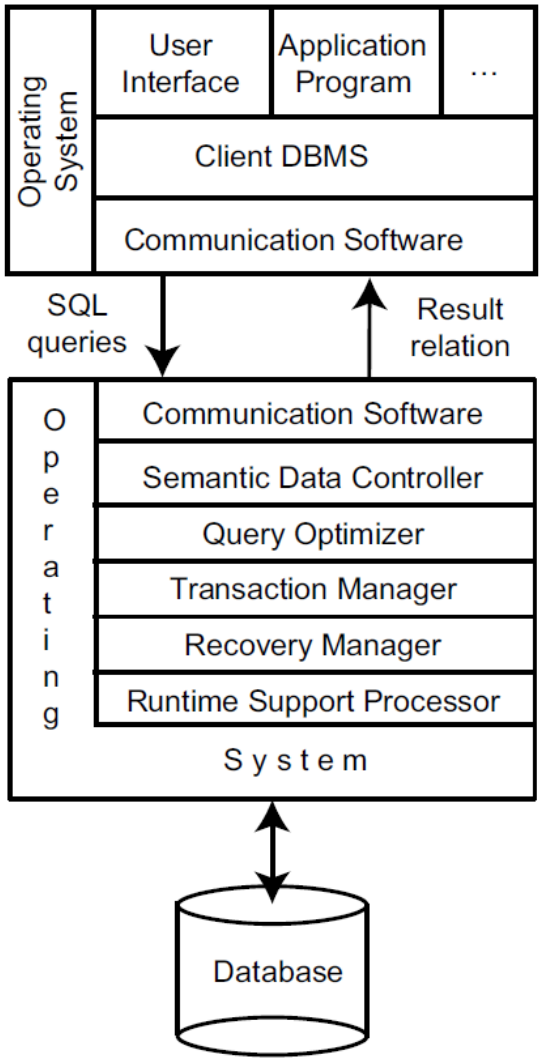


Fig. 1.11 Client/Server Reference Architecture [5]

[5] Изображение взято из [Özsu and Valduriez, 2009]

Трехуровневая архитектура I

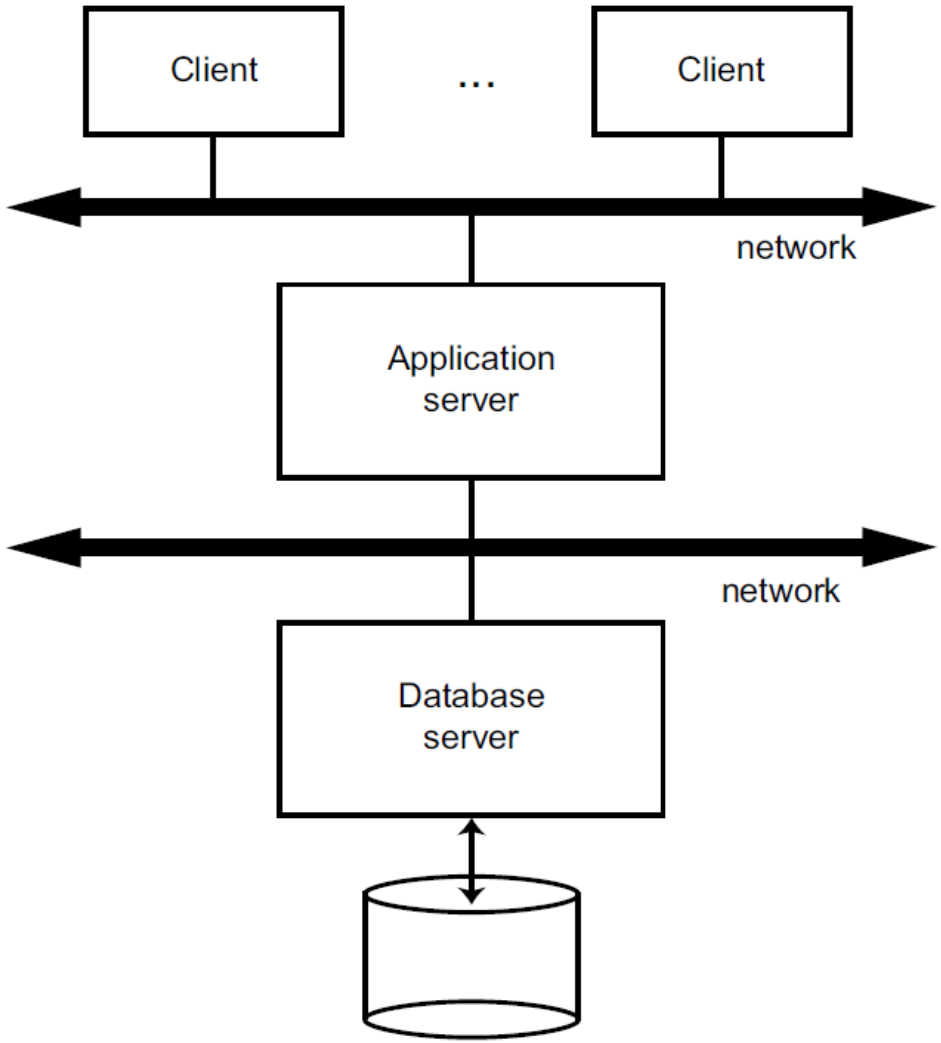


Fig. 1.12 Database Server Approach

6

[6] Изображение взято из [Özsu and Valduriez, 2009]

Трехуровневая архитектура II

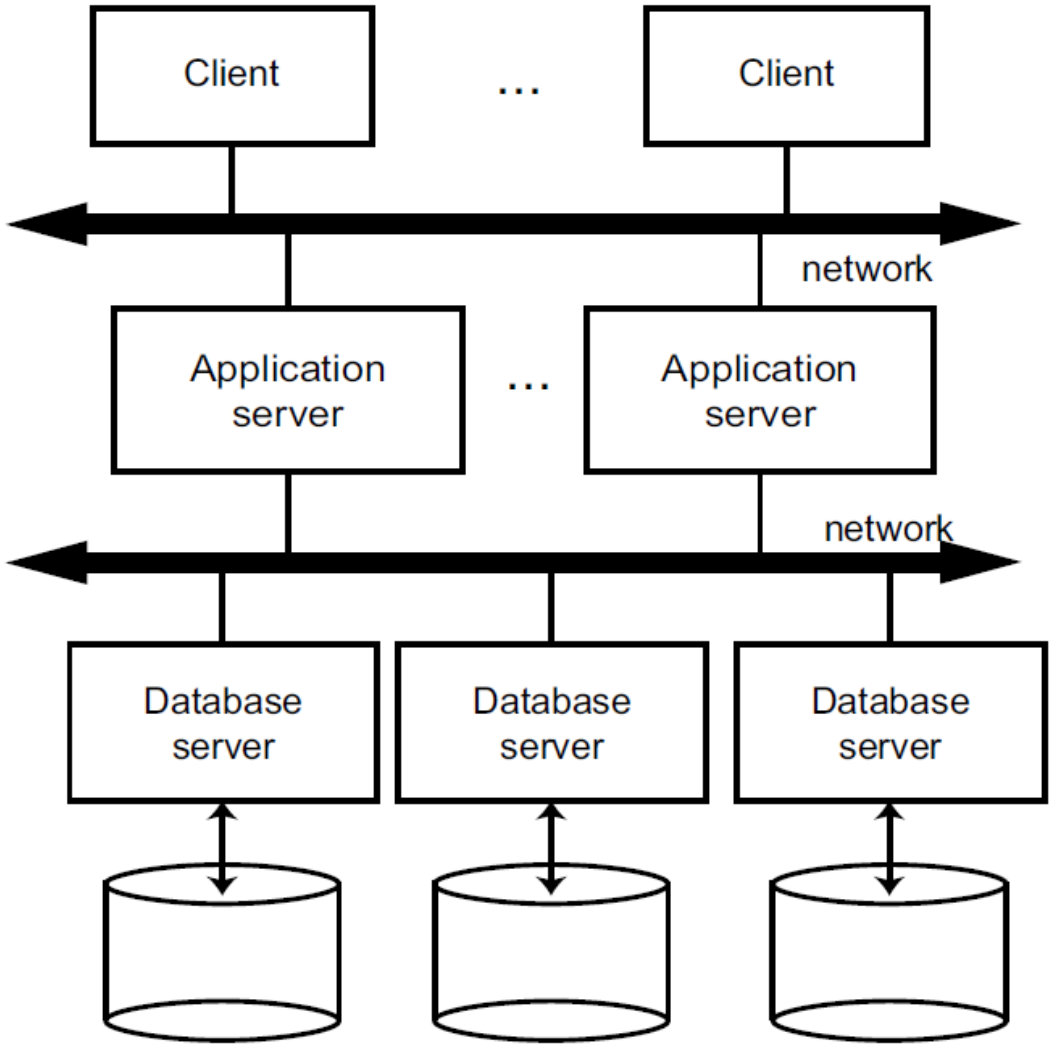


Fig. 1.13 Distributed Database Servers

[7]

[7] Изображение взято из [Özsu and Valduriez, 2009]

Основные типы РСУБД II

Типы:

- (A0, D2, H0) Peer-to-Peer РСУБД;
  - Два периода популярности: до 90-х и в нулевые;
  - Машины не различаются между собой;
  - local internal schema (LIS) — физическая организация на узлах может быть различной, поэтому ее надо описывать (на всех узлах);
  - local conceptual schema (LCS) — описывает фрагментирование и реплицирование на узле, объединение дает GCS;
  - global conceptual schema (GCS) — хранит общий вид логической структуры на всех узлах;
  - external schema (ES) — для пользовательского доступа.

Пояснение про схемы

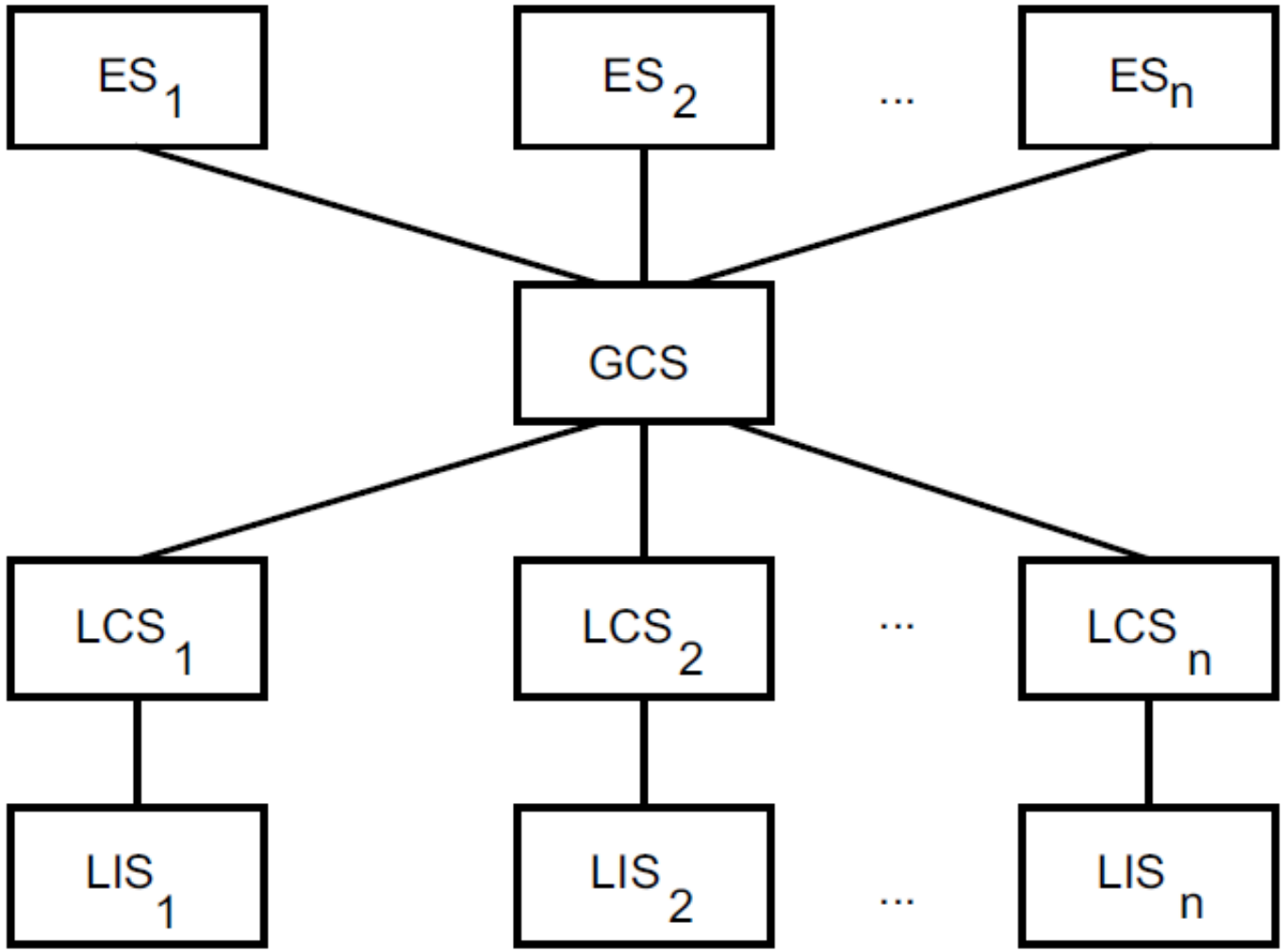
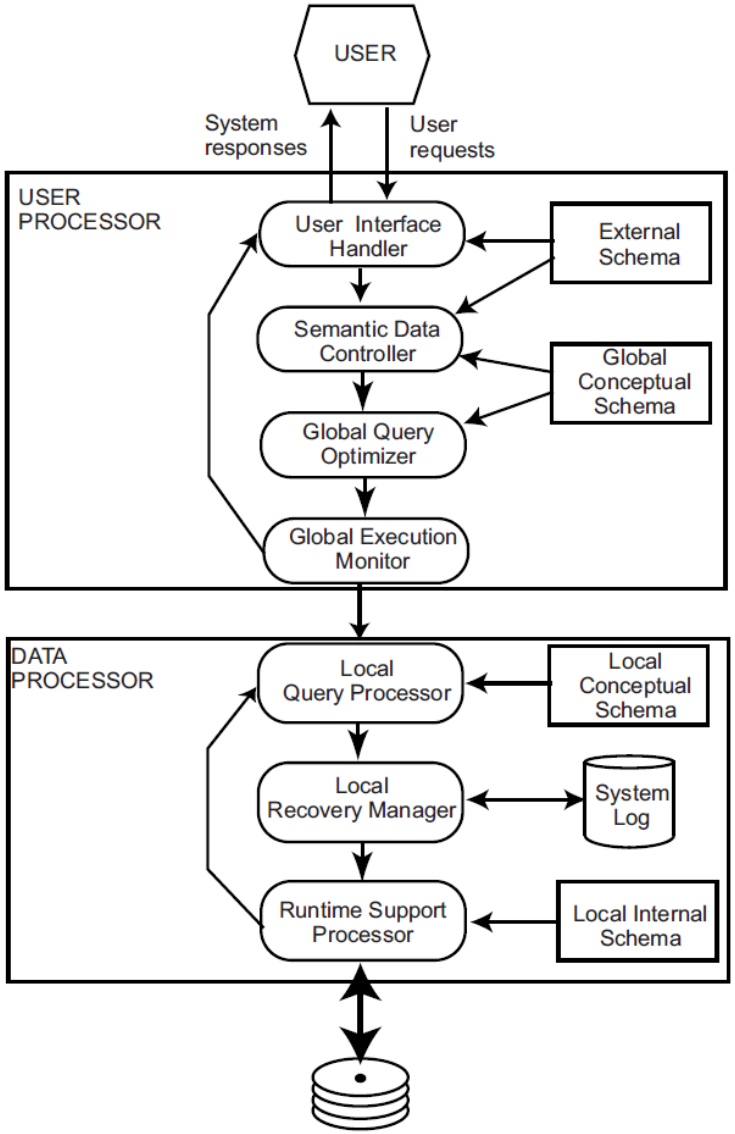


Fig. 1.14 Distributed Database Reference Architecture

8

[8] Изображение взято из [Özsu and Valduriez, 2009]

Peer-to-Peer РСУБД



9

[9] Изображение взято из [Özsu and Valduriez, 2009]

Основные типы РСУБД III

Типы:

- (A2, D2, H1) (p2p) гетерогенная мультибаза (интеграционная система, гетерогенная система).
  - полностью автономны, отдельные СУБД друг о друге не знают;
  - другое понятие глобальной базы данных, подмножество: то, что отдельные базы готовы выложить в общий доступ;
  - GCS $= \cup$ ES участников или же
    GCS $= \cup$ часть LCS участников;
  - часто реализуются посредством mediator или middleware;
  - иногда надо писать wrapper (обертку).

Устройство мультибазы

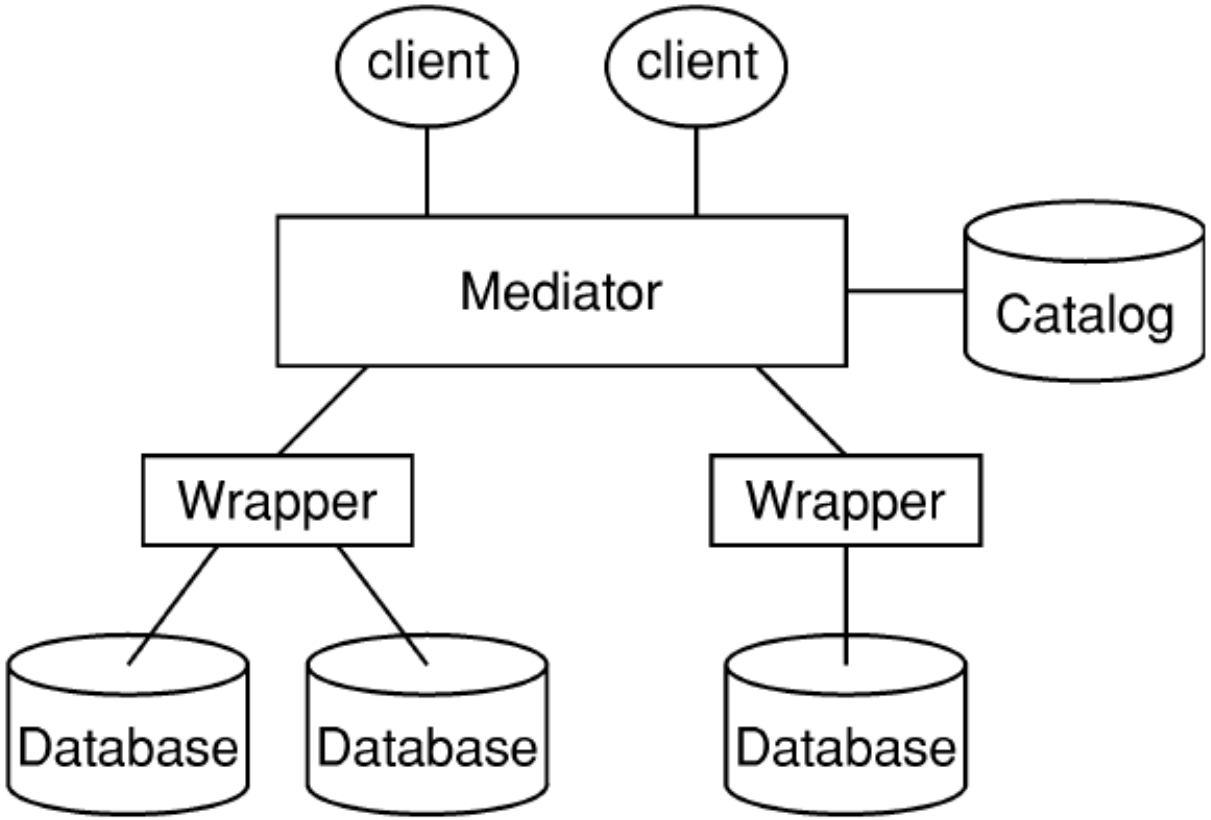


Fig. 12. Wrapper architecture of heterogeneous databases.

10

[10] Изображение взято из [Kossmann, 2000]

Ссылки I


Distributed DBMS. Sameh Elnikety. Encyclopedia of Database Systems. Ling Liu and M. Tamer Özsu (eds), p. 896–899. Springer US, 2009. http://dx.doi.org/10.1007/978-0-387-39940-9_654


Distributed Database Systems. Kian-Lee Tan. Encyclopedia of Database Systems. Ling Liu and M. Tamer Özsu (eds), p. 894–896. Springer US, 2009. http://dx.doi.org/10.1007/978-0-387-39940-9_701


Özsu M.T. and Valduriez P. Principles of Distributed Database Systems, 3rd ed. Prentice-Hall, 2011.


Donald Kossmann. 2000. The state of the art in distributed query processing. ACM Comput. Surv. 32, 4 (December 2000), 422–469. DOI=http://dx.doi.org/10.1145/371578.371598