Тема 3. *Моделювання технічної архітектури підприємства.*

Мета

Розібрати основні принципи та підходи до проектування архітектури ІС і їх технічну реалізацію.

Архітектурні стилі

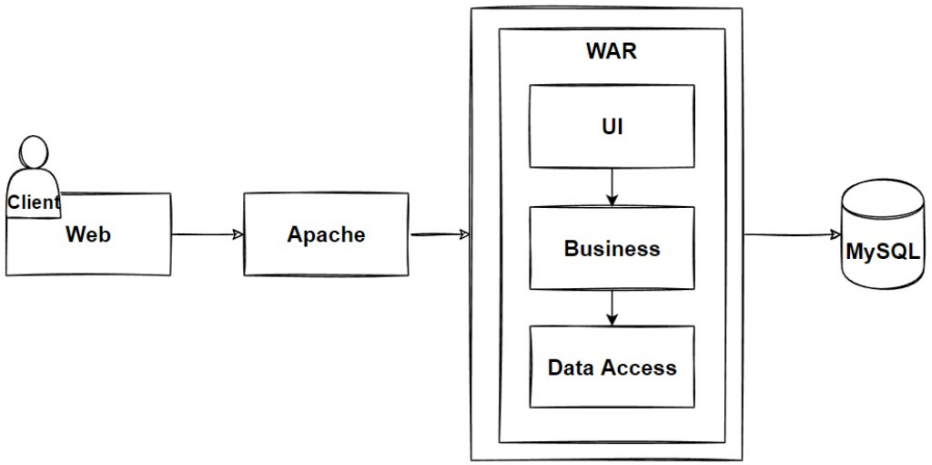
Стиль архітектури програмного забезпечення стосується високорівневої структурної організації, яка визначає загальну організацію системи, вказуючи, як організовані компоненти, як вони взаємодіють та обмеження на ці взаємодії.

Стилі архітектури зазвичай включають словник типів компонентів та з'єднувачів, а також семантичні моделі для інтерпретації властивостей системи. Ці стилі представляють найгрубіший рівень організації системи.

Основні стилі:

- Monolithic Architecture (Монолітна)
- Layered Architecture (Багатошарова, N-Tier Architecture)
- Microkernel/Plug-in (Мікроядерна)
- Service-Based Architecture, Service-Oriented Architecture
- Microservices (Мікросервісна)
- Event-Driven Architecture (Подієва)

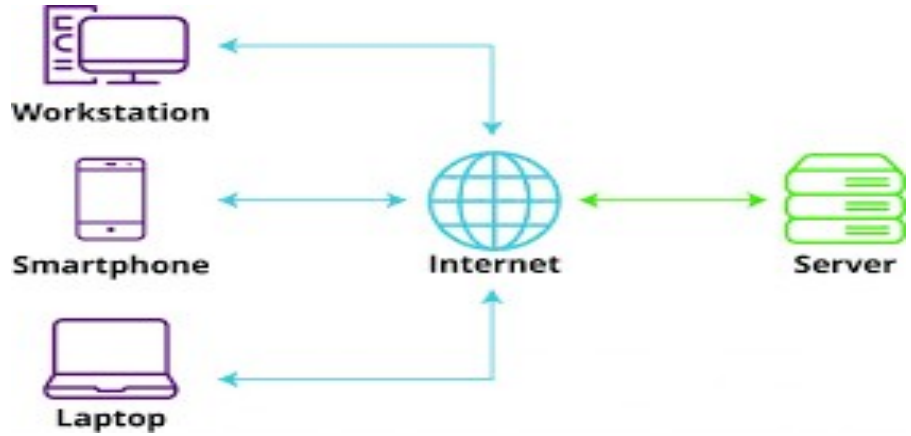
Монолітна архітектура



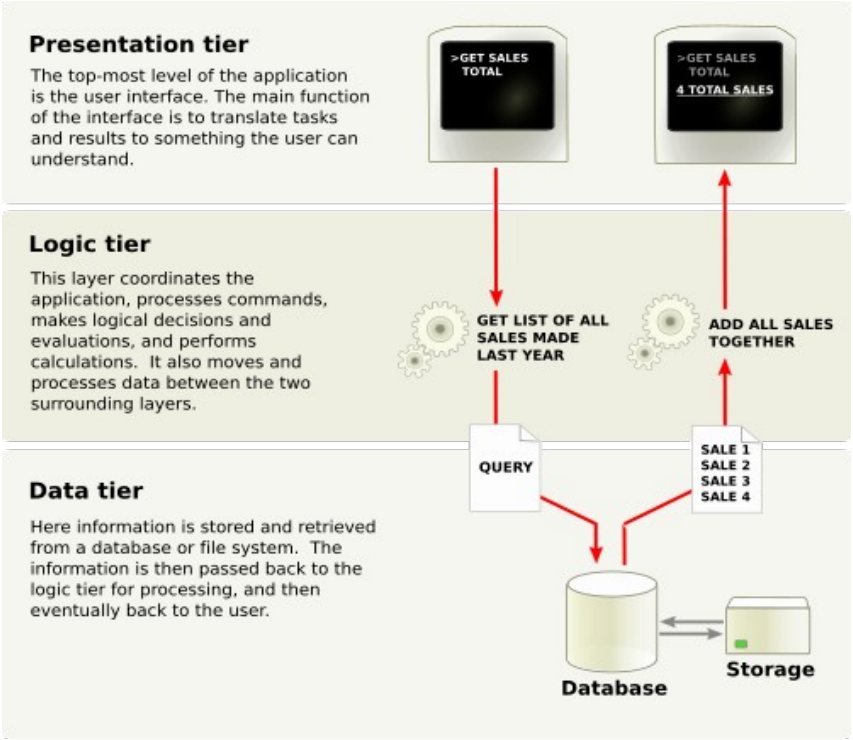
Увесь застосунок — одна збірка/процес із єдиною базою даних. Просте розгортання, прямі виклики методів, транзакційна цілісність.

Клієнт-серверна архітектура

- Взаємодії клієнта і сервера.
- Недоліки: залежність від серверу.

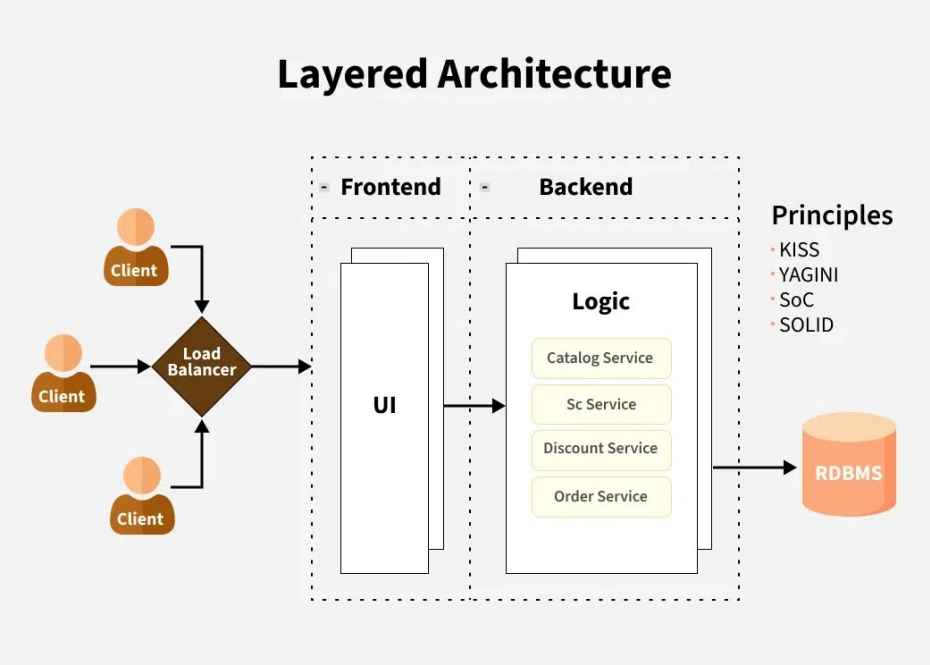


Багатошарова архітектура



- **Презентаційний рівень:**
  - Інтерфейси для користувачів (UI/UX)
- **Рівень бізнес-логіки:**
  - Реалізація бізнес-процесів
- **Рівень доступу до даних:**
  - Управління базами даних
- **Інтеграційний рівень**

Багатошарова архітектура



Поділ на шари (презентація, застосунок/домен, дані; інколи інтеграційний).

Чіткі межі відповідальностей.

Презентаційний рівень

- Інтерфейси користувача (UI/UX).
- Взаємодія з кінцевими користувачами.
- Технології: веб-інтерфейси, мобільні додатки.

Рівень бізнес-логіки

- Відповідає за реалізацію бізнес-процесів.
- Впровадження правил та алгоритмів.
- Визначення архітектурного стилю

Рівень доступу до даних

- Управління базами даних (SQL, NoSQL).
- Швидкий доступ до інформації та оптимізація запитів.
- Індексація та кешування даних.

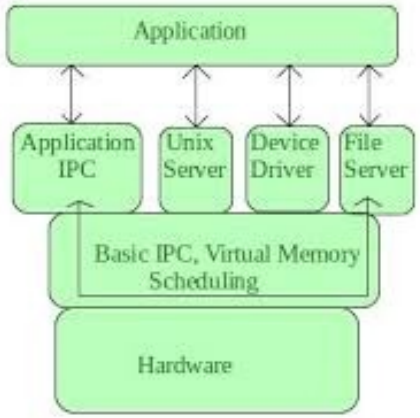
Інтеграційний рівень

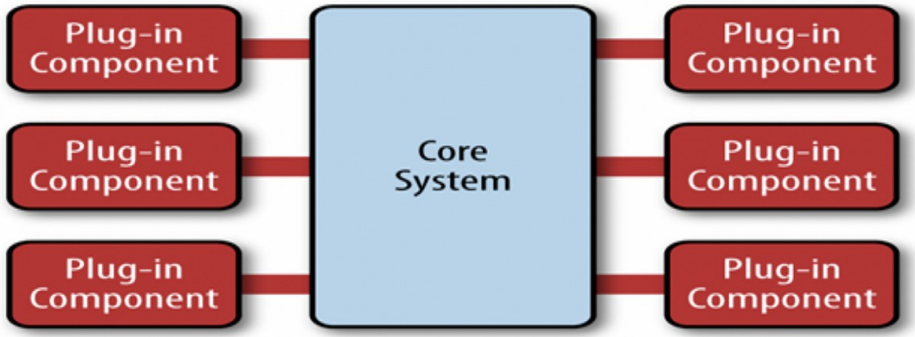
Інтеграційний рівень

- API для взаємодії з іншими системами.
- Використання веб-сервісів (REST, SOAP).
- Інтеграція з хмарними платформами.

Мікроядерна (Microkernel / Plug-in)

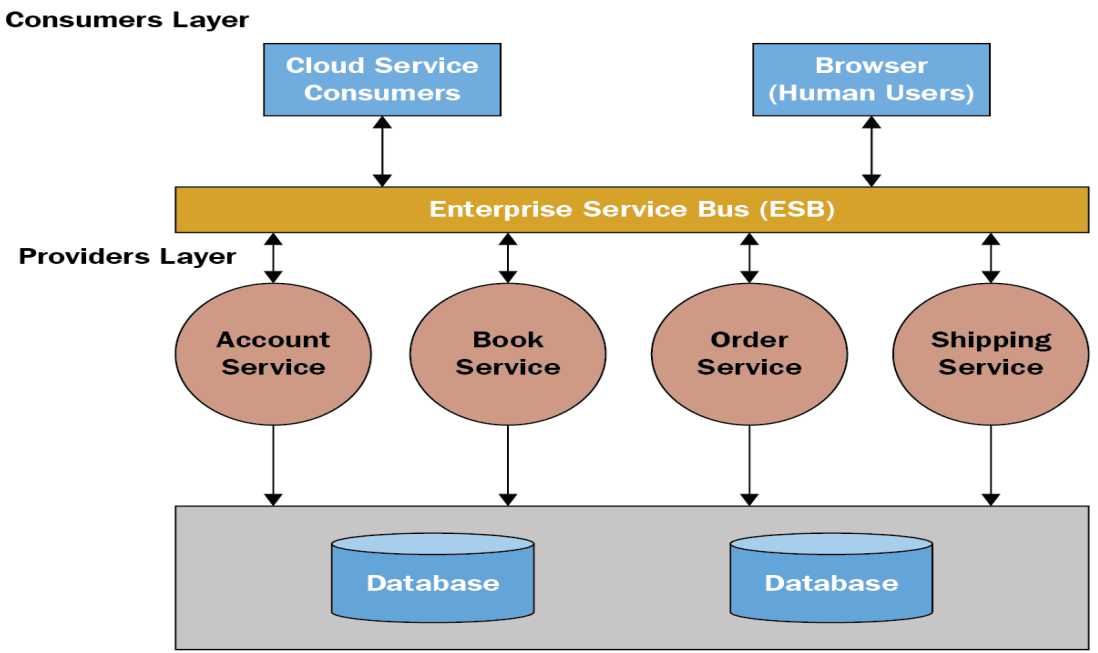
Стабільне «ядро» + незалежні плагіни/модулі, що підключаються стандартними портами/інтерфейсами



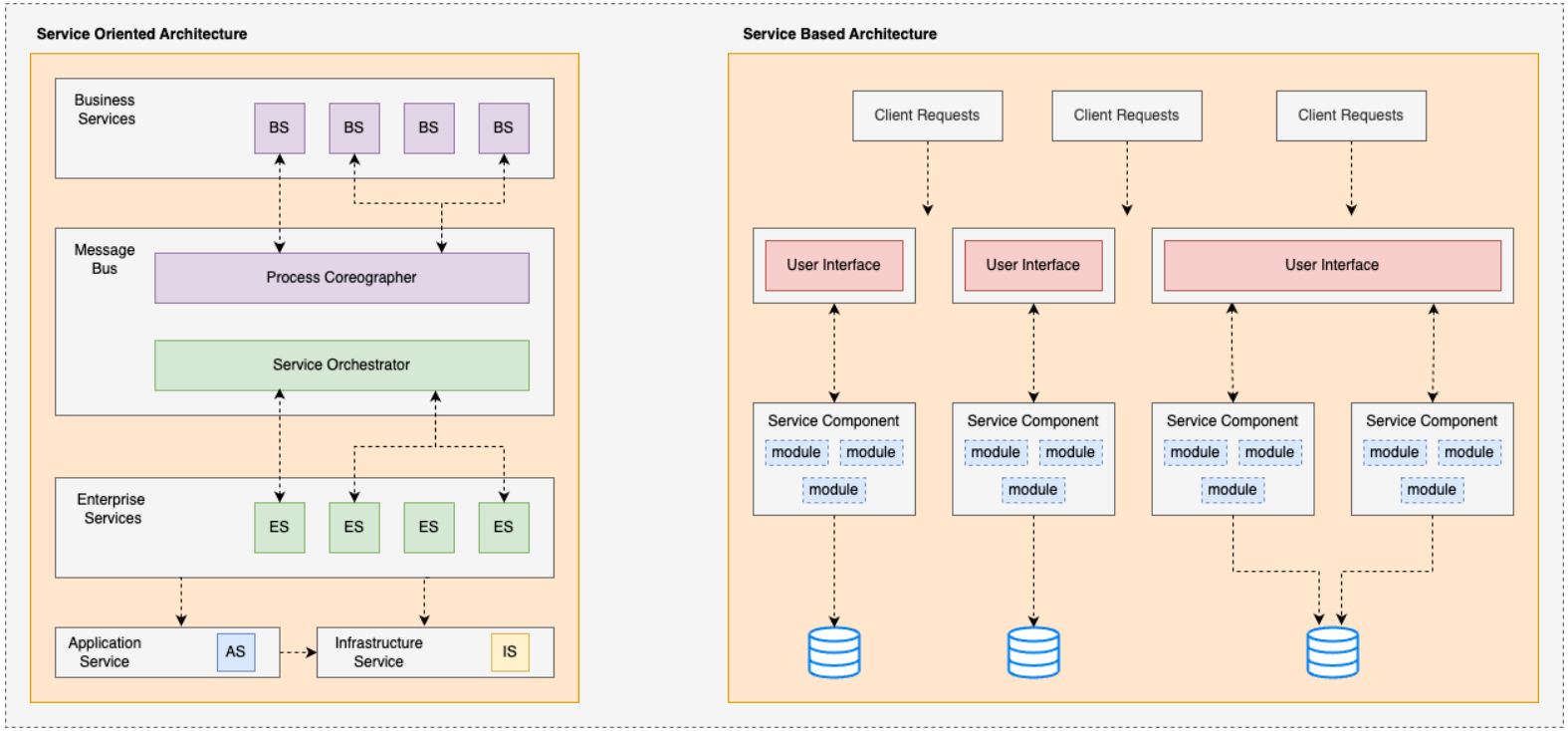


Сервіс-орієнтована архітектура (SOA)

- Сервіси як основні компоненти.
- Взаємодія через стандартизовані протоколи.
- Відмінності від мікросервісів.

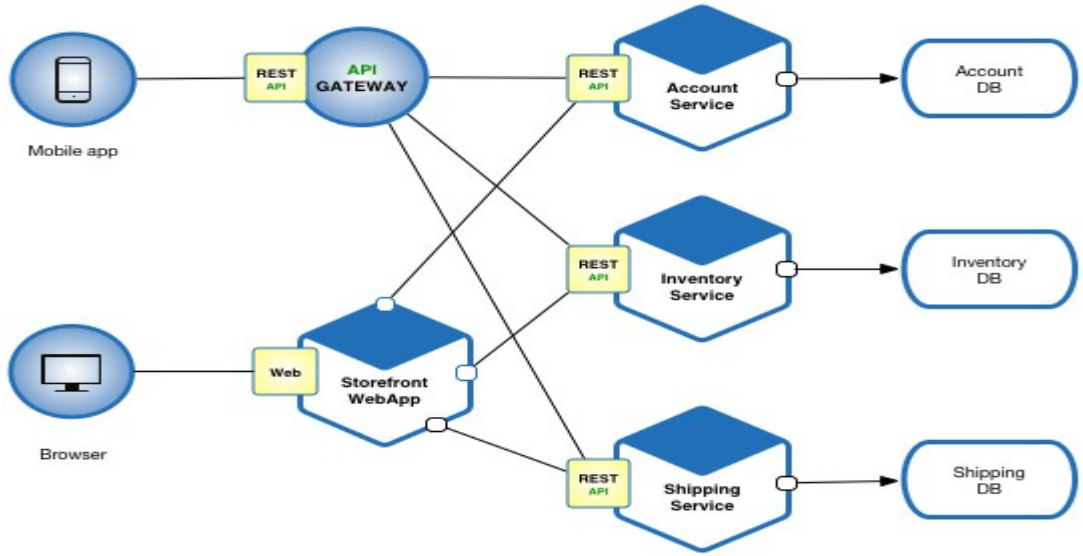


SOA / SBA


Service Oriented Architecture
Business Services
BS
BS
BS
BS
Message Bus
Process Coreographer
Service Orchestrator
Enterprise Services
ES
ES
ES
ES
Application Service
AS
Infrastructure Service
IS
Service Based Architecture
Client Requests
Client Requests
Client Requests
User Interface
User Interface
User Interface
Service Component
module
module
module
Service Component
module
module
module
Service Component
module
module
module
Service Component
module
module
module

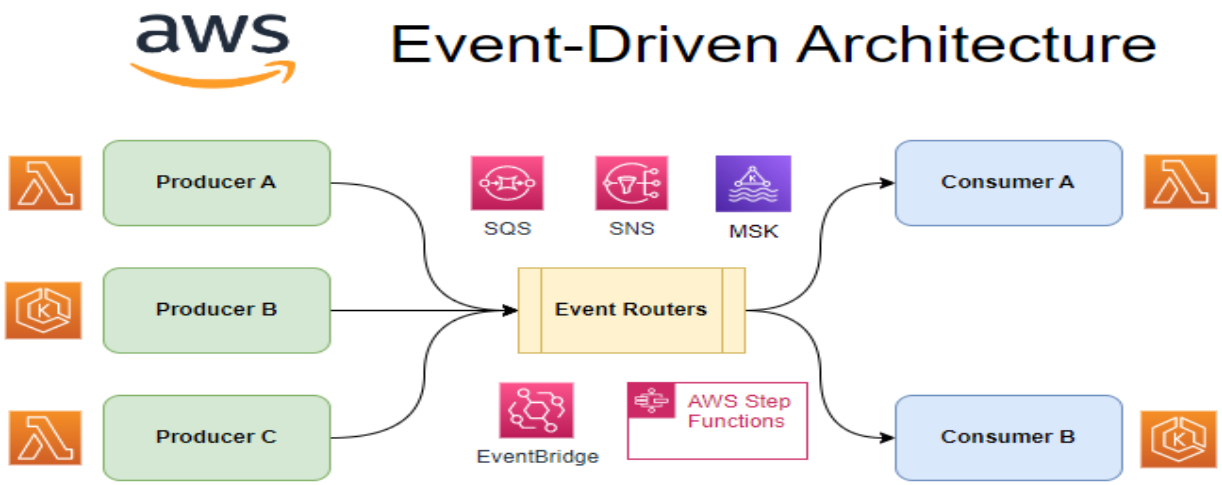
Мікросервісна архітектура

- Розподіл системи на мікросервіси.
- Оркестрація та контейнеризація (Docker, Kubernetes).
- Переваги: гнучкість та автономність сервісів.



Архітектура на основі подій

- Використання подій для запуску дій.
- Реалізація через черги повідомлень
- Переваги в масштабованості.



Вибір архітектурного підходу

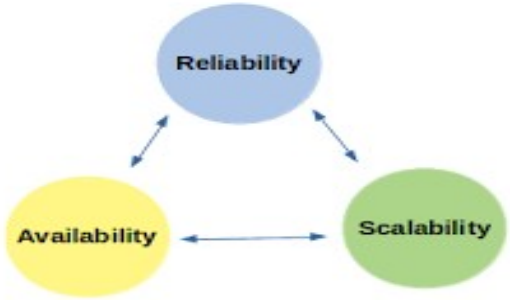
Фактори вибору архітектури:

- Масштаб проекту
- Вимоги до продуктивності
- Інфраструктура

Нефункціональні вимоги

Ключові вимоги:

- Масштабованість
- Доступність
- Надійність



Інструменти для проектування

- UML діаграми
- TOGAF, Zachman Framework
- DevOps інструменти

DevOps

- Впровадження практик DevOps.
- (Слово DevOps — це комбінація термінів «розробка» та «операції», що означає спільний підхід до завдань, які виконують команда розробників та команда IT-операцій)
- CI/CD: інтеграція та доставка коду.
- Використання контейнерів та хмарних сервісів.

Дякую за увагу!