Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

**Звіт**

Тема: “Створення базової архітектури проекту”

Виконала: Перевірено:

студентка групи ІТ-71 асист. кафедри АУТС

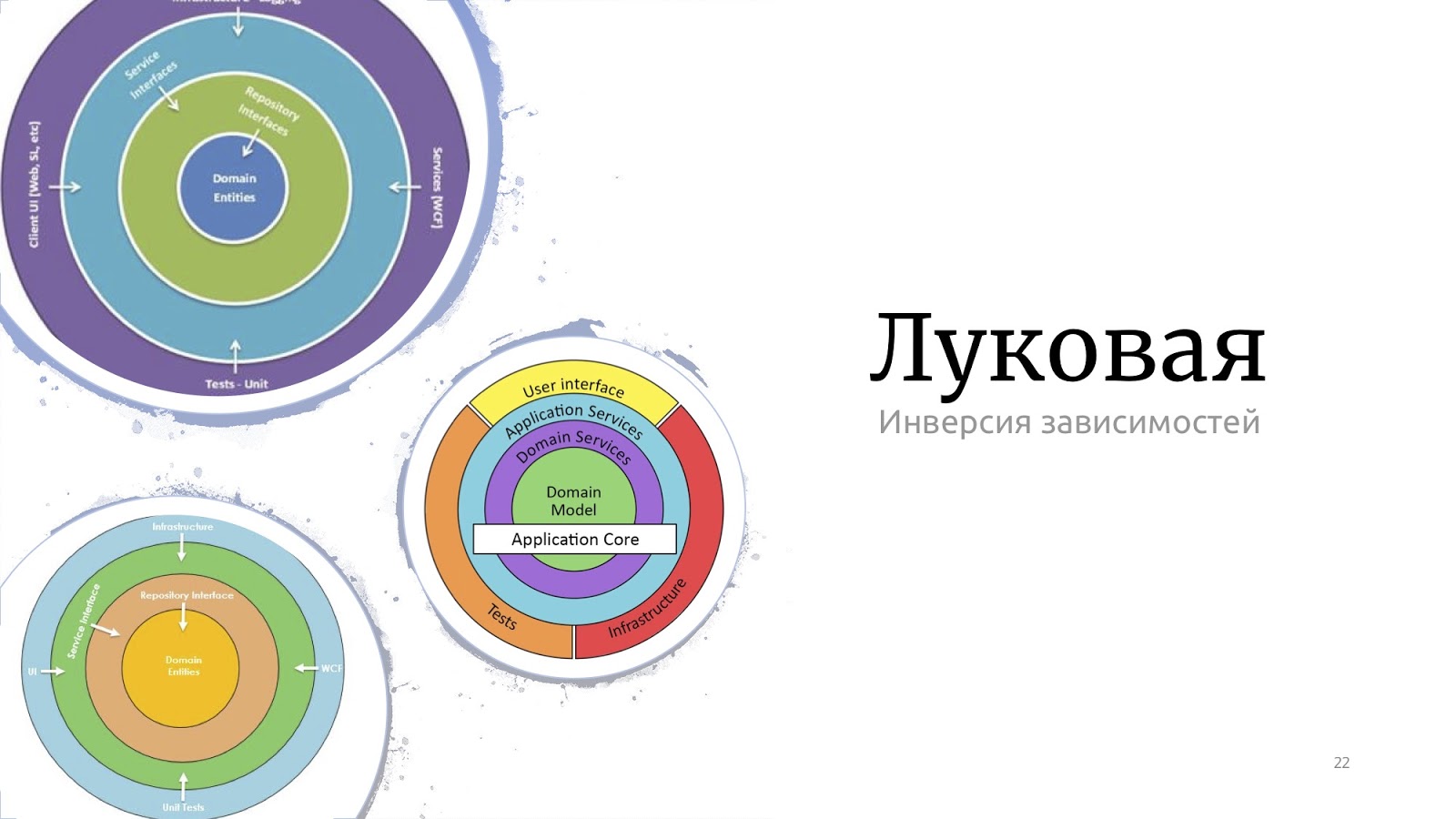
Новиков Данило Галушко Дмитро

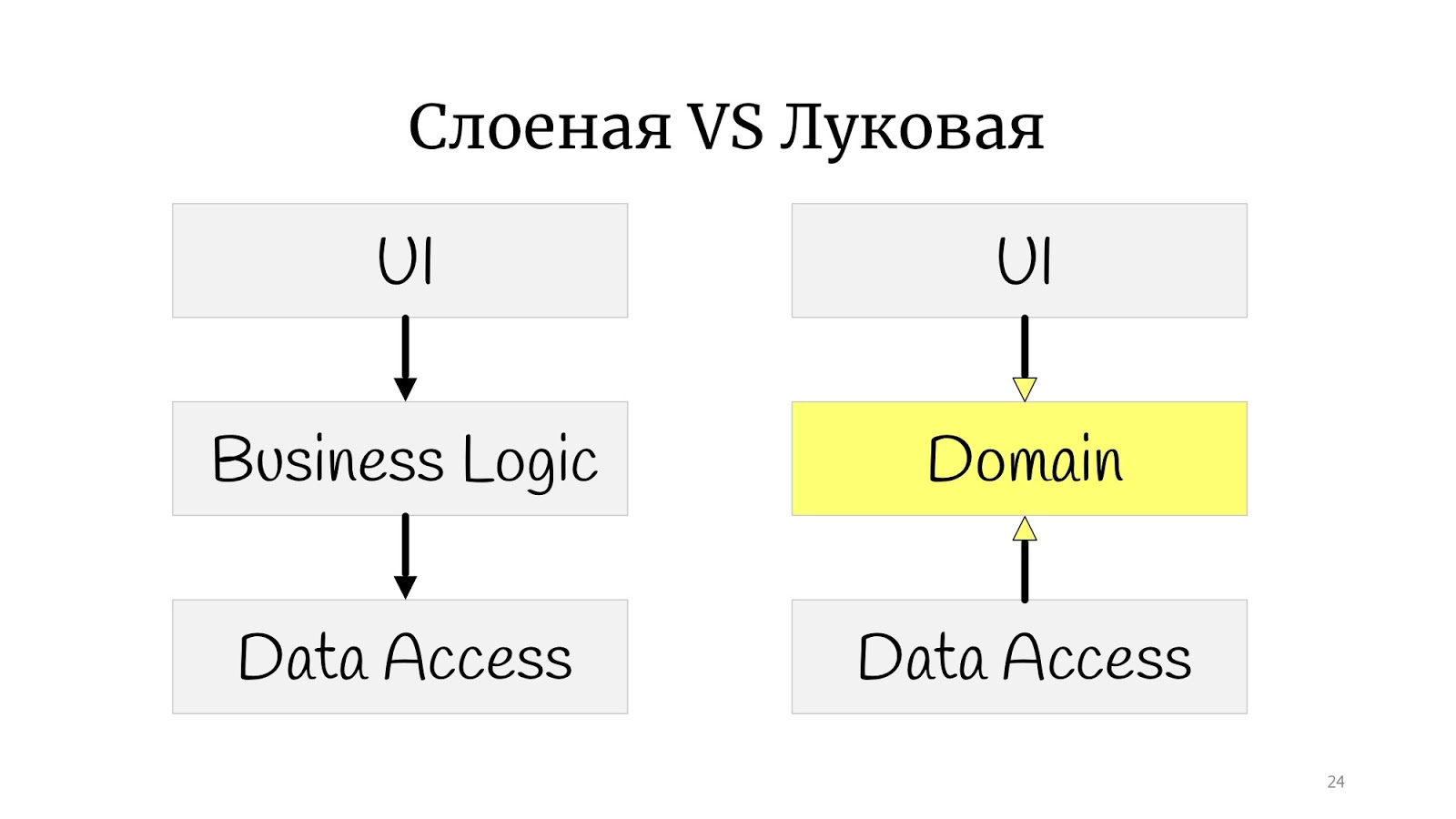
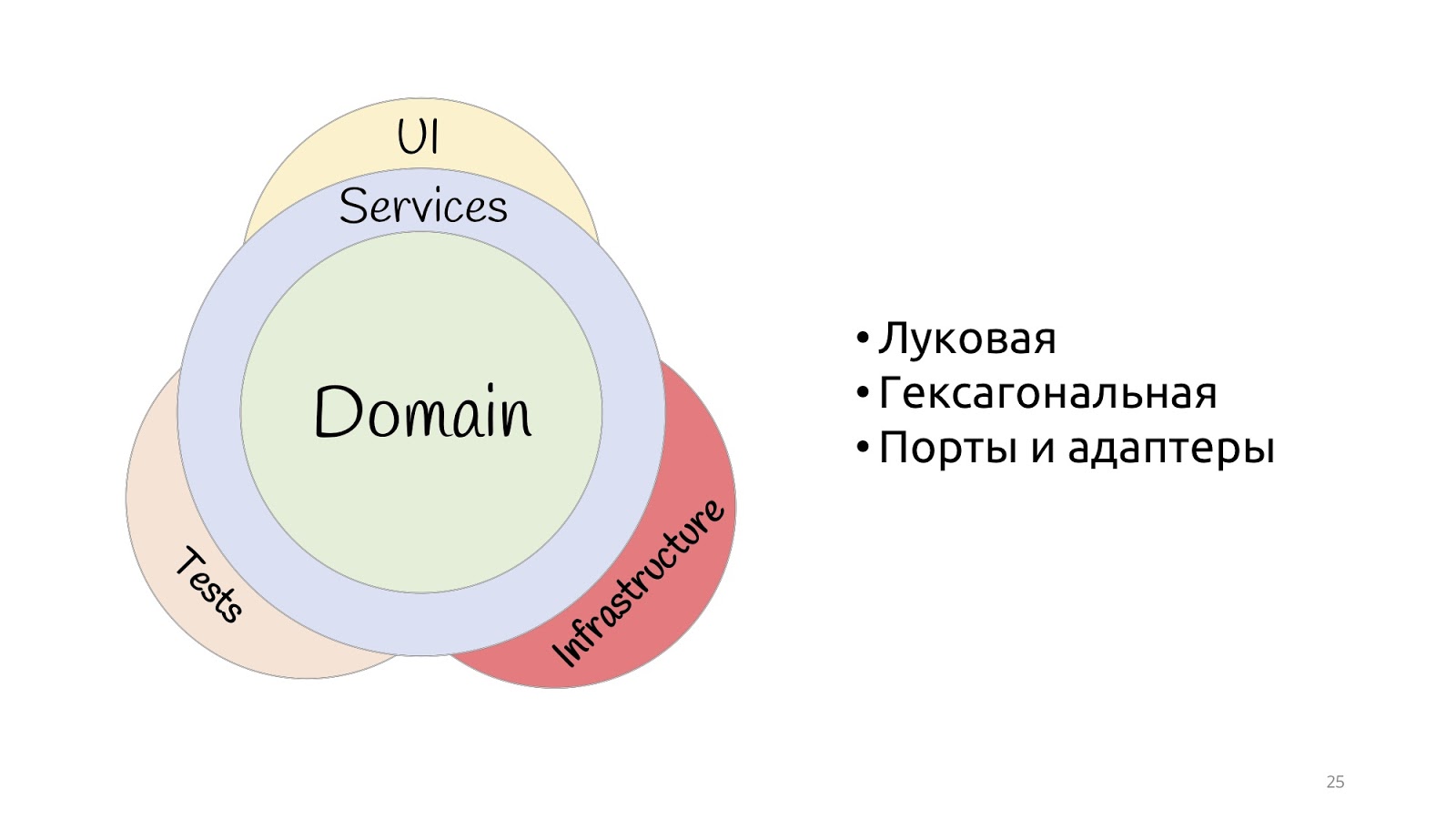
Михайлович Олександрович

Київ 2020

Замість дефолтної «листкової» архітектури було обрано лукову:



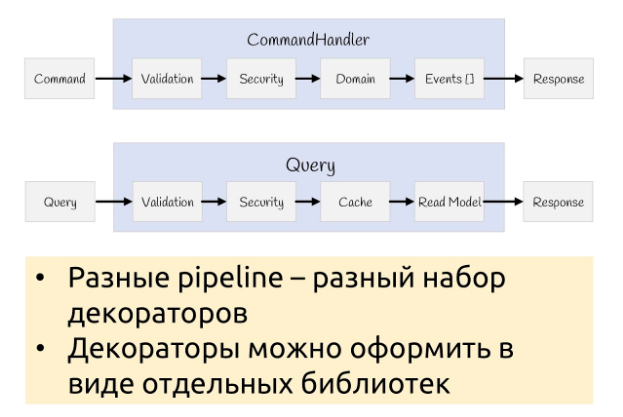


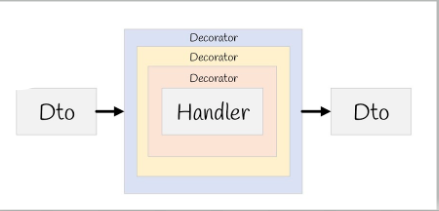
За основу було обрано спрощений варіант CQRS(без dispatcher та DDD) - command-query responsibility segregation (CQRS) — це архітекрутний патерн, який вказує що метод повинен бути або командою, яка виконує якусь дію, або запитом, що повертає дані, але не одночасно і тим, і іншим. Інакше кажучи, задавання питання не повинно змінювати відповідь. Цей підхід дає нам наступні переваги:

* Самостійне масштабування. CQRS дозволяє навантажувати читання та запис масштабів самостійно масштабувати, і це може призвести до меншої кількості блокувань блокування.
* Оптимізовані схеми даних. Сторона читання може використовувати схему, оптимізовану для запитів, тоді як сторона запису використовує схему, оптимізовану для оновлень.
* Безпека. Простіше переконатися, що лише ті потрібні об'єкти домену виконують запис у дані.
* Розділення проблем. Розмежування сторін читання і запису може призвести до отримання більш гнучких і гнучких моделей. Більшість складної логіки бізнесу переходить у модель запису. Прочитана модель може бути відносно простою.
* Простіші запити. Зберігаючи матеріалізований вигляд у читаній базі даних, програма може уникнути складних приєднань під час запитів.

Також обробку усіх базових речей було винесено у декоратори для спрощення та уніфікації роботи з базовими методами/додаванням нових речей. Тобто тепер звичайний запит є собой pipeline з декораторів та має наступний (приблизний) вигляд:



Загальний же підхід обробки запитів має наступний вигляд:



Загалом ми маємо «матрьошку» з декораторів, які можуть бути повторно перевикористанні(окрім спецефічного для кожного метода).

Джерела:

1. Скриншоти та ідея архітекрути були взяті зі статті(текстової інтерпретації його доклада з конференції) Максима Аршинова «Быстрорастворимое проэктирование» - <https://habr.com/ru/company/jugru/blog/447308/>;

Інші цікаві та корисні матеріли по даному напряму:

1. «CQRS. Факты и заблуждения» - <https://habr.com/ru/post/347908/>;
2. «Как мы попробовали DDD, CQRS и Event Sourcing и какие выводы сделали» - <https://habr.com/ru/post/313110/>;
3. «DDD, Hexagonal, Onion, Clean, CQRS… как я собрал всё это вместе» - <https://habr.com/ru/post/427739/>;
4. «CQRS – Simple architecture» - <https://kariera.future-processing.pl/blog/cqrs-simple-architecture/>.