Архитектура производственного DLMS-конфигуратора:

деление системы на уровни (3 компонента)

Правила хорошей архитектуры («Чистая архитектура», Р. Мартин – глава 17, с. 172, цитата):

«Графический интерфейс не имеет значения для бизнес-правил, поэтому между ними нужно

провести границу. База данных не имеет значения для графического интерфейса, поэтому между ними нужно провести границу. База данных не имеет значения для бизнес-правил, поэтому между ними нужно провести границу.»

|  |  |
| --- | --- |
| № | Уровень (компонент) |
| **1** | **Графический интерфейс UI (зависимый компонент).**  Содержит: виджеты, логику UX, модели данных для доступа к данным, верхне-уровневые функции загрузки данных (не раскрывающие структур данных), верхне-уровневые функции применения бизнес-правил (не раскрывающие правил).  Не содержит: бизнес-правила, данные, конкретный код загрузки/сохранения данных.  Назначение модели данных (из MVC) – интерфейс для доступа к данным; не обязательно хранение данных в ней, может даже очень нежелательно. |
| **2** | **Бизнес-правила (независимый компонент).** |
| 2.1 | **Бизнес-правила, связанные с приложением.** Проверка полей ввода. |
| 2.2 | **Бизнес-правила, связанные с предметной областью.**  Содержит: правила для адресных плат, стендов, процедур поверки, калибровки (понятные непрограммистам).  Бизнес-правила:   1. Обмен данными между системой и счетчиками должен производиться через адресные платы. 2. Подача/снятие напряжения на счетчики должны производиться через стенд. 3. Процедуры поверки должны храниться в системе (программе). 4. Данные поверок должны храниться в файлах. 5. Процедуры калибровки должны храниться в системе. |
| 2.3 | **Требования к системе – д**олжны быть созданы на основе бизнес-правил.  Требования.   1. Адресные платы и их логика должны быть изолированы, изменения их логики не должно изменять классы UI и классы работы с данными (DataStore, Loaders, Savers). 2. Стенды и их логика должны быть изолированы и также не влиять на классы UI и классы работы с данными. 3. Процедура поверки (правило) и результаты поверки (данные) должны быть разделены. Отображение результатов поверки UI (модель из MVC, виджет) и результаты поверки (данные) должны быть разделены. Процедура поверки (правило) и UI должны быть разделены. |
| **3** | **База данных (зависимый компонент).**  Содержит базу данных и классы доступа к данным. База данных: файлы с данными, данные в ОЗУ. Классы доступа к данным: код доступа к данным, код загрузки данных, код сохранения данных.  Бизнес-правила не должны зависеть от типа базы данных. Конкретная база данных должна быть скрыта за интерфейсом. Бизнес-правила не должны зависеть от UI.  Компоненты UI и базы данных должны зависеть от бизнес-правил. Однако бизнес-правила не должны явно содержаться в этих компонентах и также должны быть изолированы, чтобы изменения в UI или DataStore не повлияли на правила. |

**Вывод.** Чтобы разобраться детально и на что-то опираться (кроме SOLID) при обсуждении «как должно быть спроектировано» надо определить принадлежность каждой сущности (класса, структуры и пр.) к одному из трех компонентов.