

Points of View: ключ к  
общению разработчиков,  
архитекторов, QAs и PMs.  
Видим качество за  
диаграммами.

Евгений Кривошеев,  
Scrumtrek,  
[ekrivosheyev@scrumtrek.ru](mailto:ekrivosheyev@scrumtrek.ru)

## **В какой момент возникают проблемы с качеством**

- Моменты принятия инженерных решений
- Моменты оценок своих решений инженерами
- Моменты общения ключевых ролей

# Как должны приниматься инженерные решения

- Дизайн как компромисс
- Обоснованность решений через требования

В реальности: «Все плохо.  
Но уже *поздно.*»

# Почему QA видит проблемы лучше DEV

- Понимание требований
- Фокус на внешнем качестве
- Нет привязанности к коду

## Как QA может рано увидеть проблемы

- Меппинг инженерных решений на характеристики
- Право на вето или необходимость эксперимента

# Почему РМ видит проблемы лучше DEV

- Общее видение
- Дополнительные проектные знания по ограничениям



# Как РМ может рано увидеть проблемы

- Меппинг инженерных решений на план проекта
- Право на вето или необходимость эксперимента
- Раннее разрешение внешней и внутренней неопределенностей

# Фреймворк коммуникаций

- QA и PM в архитектурной группе
- Участие в решениях как можно раньше
- «Обоснуй» → через FR и NFR

Но для этого нужно  
«перевести» архитектуру  
на язык РМ и QA

---







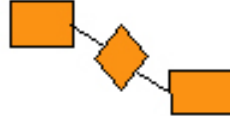
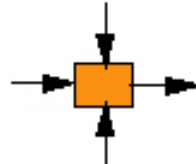

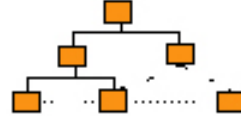
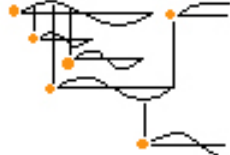
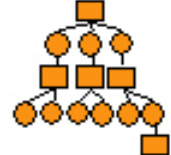
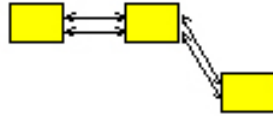
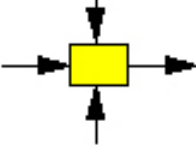
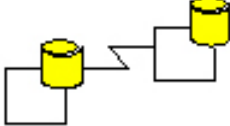
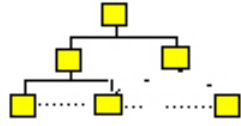

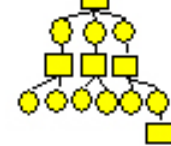
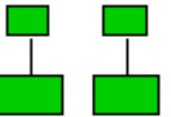
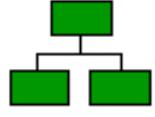
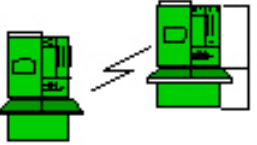

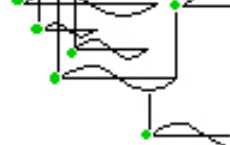
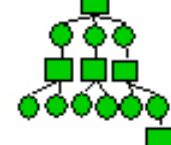






# Points of View

- «Слон», «Гиперкуб» и чудо-очки
- *Архитектура – не только для инженеров!*
- Приоритезация решений

# Архитектурные фреймворки на PoV

- 4+1
- Rozansky & Woods
- Zachman

# ENTERPRISE ARCHITECTURE - A FRAMEWORK™

|  | DATA<br><i>What</i>   | FUNCTION<br><i>How</i>   | NETWORK<br><i>Where</i>   | PEOPLE<br><i>Who</i>   | TIME<br><i>When</i>   | MOTIVATION<br><i>Why</i>  |  |
|--|---|--|---|--|---|---|--|
| SCOPE<br>(CONTEXTUAL)                        | List of Things Important to the Business<br> | List of Processes the Business Performs<br> | List of Locations in which the Business Operates<br> | List of Organizations Important to the Business<br> | List of Events Significant to the Business<br> | List of Business Goals/Strat<br> | SCOPE<br>(CONTEXTUAL)                        |
| <i>Planner</i>                               | ENTITY = Class of Business Thing  | Function = Class of Business Process   | Node = Major Business Location  | People = Major Organizations   | Time = Major Business Event   | Ends/Mean = Major Bus. Goal/Critical Success Factor   | <i>Planner</i>                               |
| ENTERPRISE MODEL<br>(CONCEPTUAL)             | e.g. Semantic Model<br>                      | e.g. Business Process Model<br>             | e.g. Business Logistics System<br>                   | e.g. Work Flow Model<br>                            | e.g. Master Schedule<br>                       | e.g. Business Plan<br>           | ENTERPRISE MODEL<br>(CONCEPTUAL)             |
| <i>Owner</i>                                 | Ent = Business Entity<br>ReIn = Business Relationship   | Proc = Business Process<br>I/O = Business Resources  | Node = Business Location<br>Link = Business Linkage   | People = Organization Unit<br>Work = Work Product  | Time = Business Event<br>Cycle = Business Cycle   | End = Business Objective<br>Means = Business Strategy   | <i>Owner</i>                                 |
| SYSTEM MODEL<br>(LOGICAL)                    | e.g. Logical Data Model<br>                  | e.g. Application Architecture<br>          | e.g. Distributed System Architecture<br>            | e.g. Human Interface Architecture<br>              | e.g. Processing Structure<br>                 | e.g. Business Rule Model<br>    | SYSTEM MODEL<br>(LOGICAL)                    |
| <i>Designer</i>                              | Ent = Data Entity<br>ReIn = Data Relationship   | Proc = Application Function<br>I/O = User Views  | Node = I/S Function (Processor, Storage, etc.)<br>Link = Line Characteristics   | People = Role<br>Work = Deliverable  | Time = System Event<br>Cycle = Processing Cycle   | End = Structural Assertion<br>Means = Action Assertion  | <i>Designer</i>                              |
| TECHNOLOGY MODEL<br>(PHYSICAL)               | e.g. Physical Data Model<br>               | e.g. System Design<br>                    | e.g. Technology Architecture<br>                   | e.g. Presentation Architecture<br>                | e.g. Control Structure<br>                   | e.g. Rule Design<br>           | TECHNOLOGY MODEL<br>(PHYSICAL)               |
| <i>Builder</i>                               | Ent = Segment/Table/etc.<br>ReIn = Pointer/Key/etc.   | Proc = Computer Function<br>I/O = Data Elements/Sets   | Node = Hardware/System Software<br>Link = Line Specifications   | People = User<br>Work = Screen Format  | Time = Execute<br>Cycle = Component Cycle   | End = Condition<br>Means = Action   | <i>Builder</i>                               |
| DETAILED REPRESENTATIONS<br>(OUT-OF-CONTEXT) | e.g. Data Definition<br>                   | e.g. Program<br>                          | e.g. Network Architecture<br>                      | e.g. Security Architecture<br>                    | e.g. Timing Definition<br>                   | e.g. Rule Specification<br>    | DETAILED REPRESENTATIONS<br>(OUT-OF-CONTEXT) |
| <i>Sub-Contractor</i>                        | Ent = Field<br>ReIn = Address   | Proc = Language Stmt<br>I/O = Control Block  | Node = Addresses<br>Link = Protocols  | People = Identity<br>Work = Job  | Time = Interrupt<br>Cycle = Machine Cycle   | End = Sub-condition<br>Means = Step   | <i>Sub-Contractor</i>                        |
| FUNCTIONING ENTERPRISE                       | e.g. DATA   | e.g. FUNCTION  | e.g. NETWORK  | e.g. ORGANIZATION  | e.g. SCHEDULE   | e.g. STRATEGY   | FUNCTIONING ENTERPRISE                       |

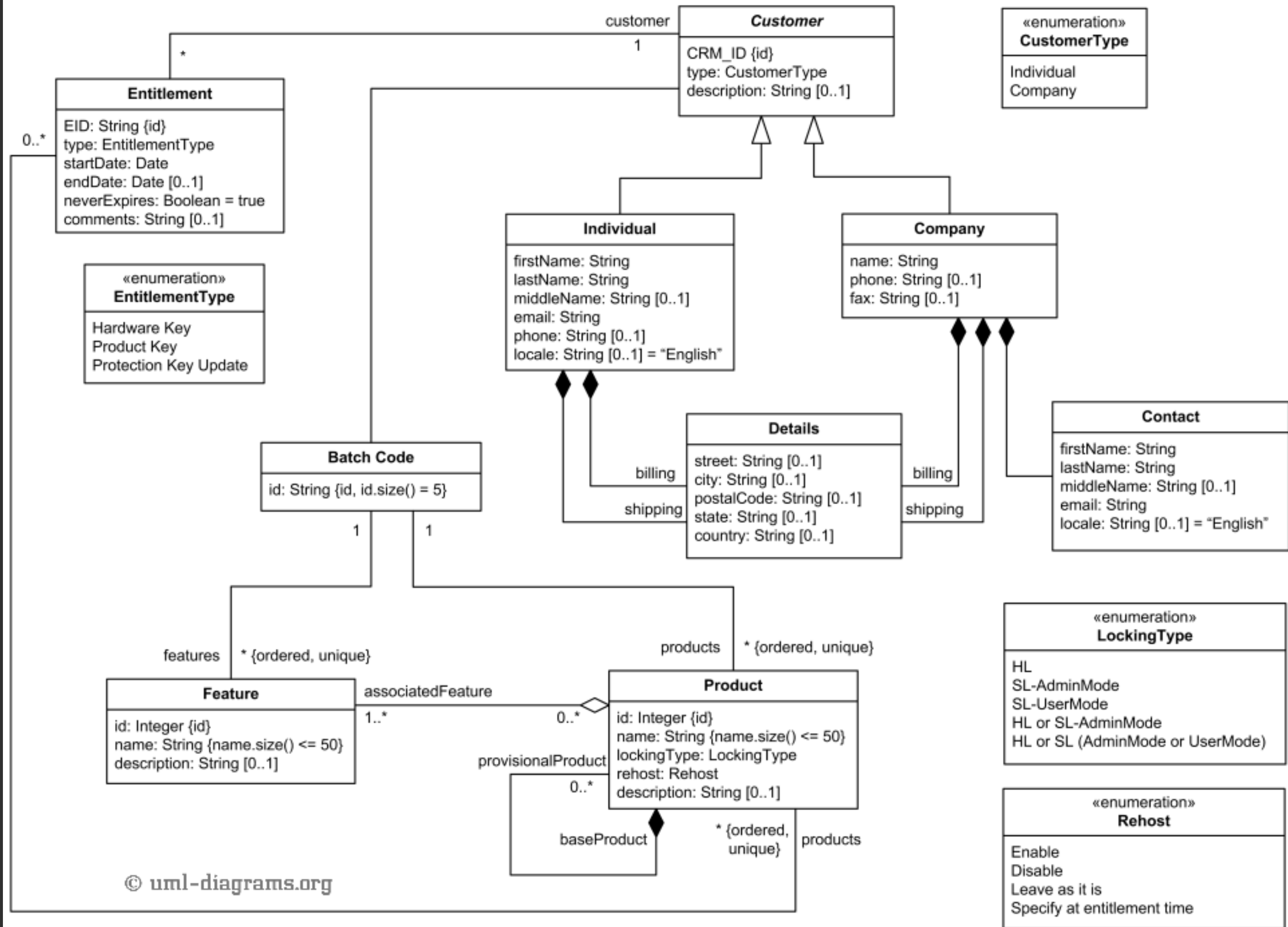
## **QA накапливает экспертизу в разных PoV**

- Объем знаний в каждой PoV
- Видеть качество за картинками

# Шаблон №1: ВИДИМ качество за UML Class Diagram

---



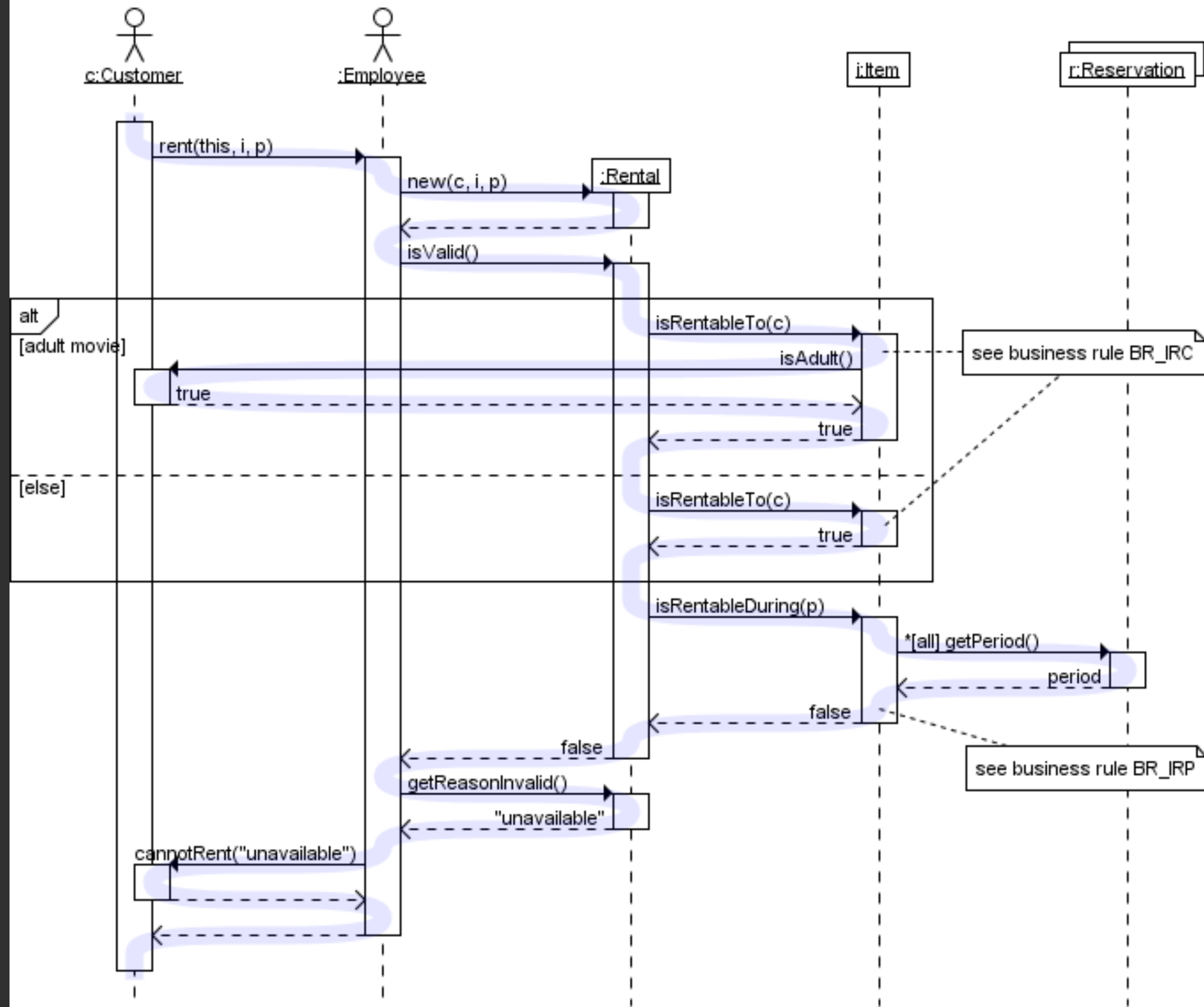


# Диагноз по Class Diagram → NFR

- Инкапсуляция
- Точки расширяемости
- Ассоциации и зависимости
- API
- State

# Шаблон №2: ВИДИМ качество за UML Sequence Diagram

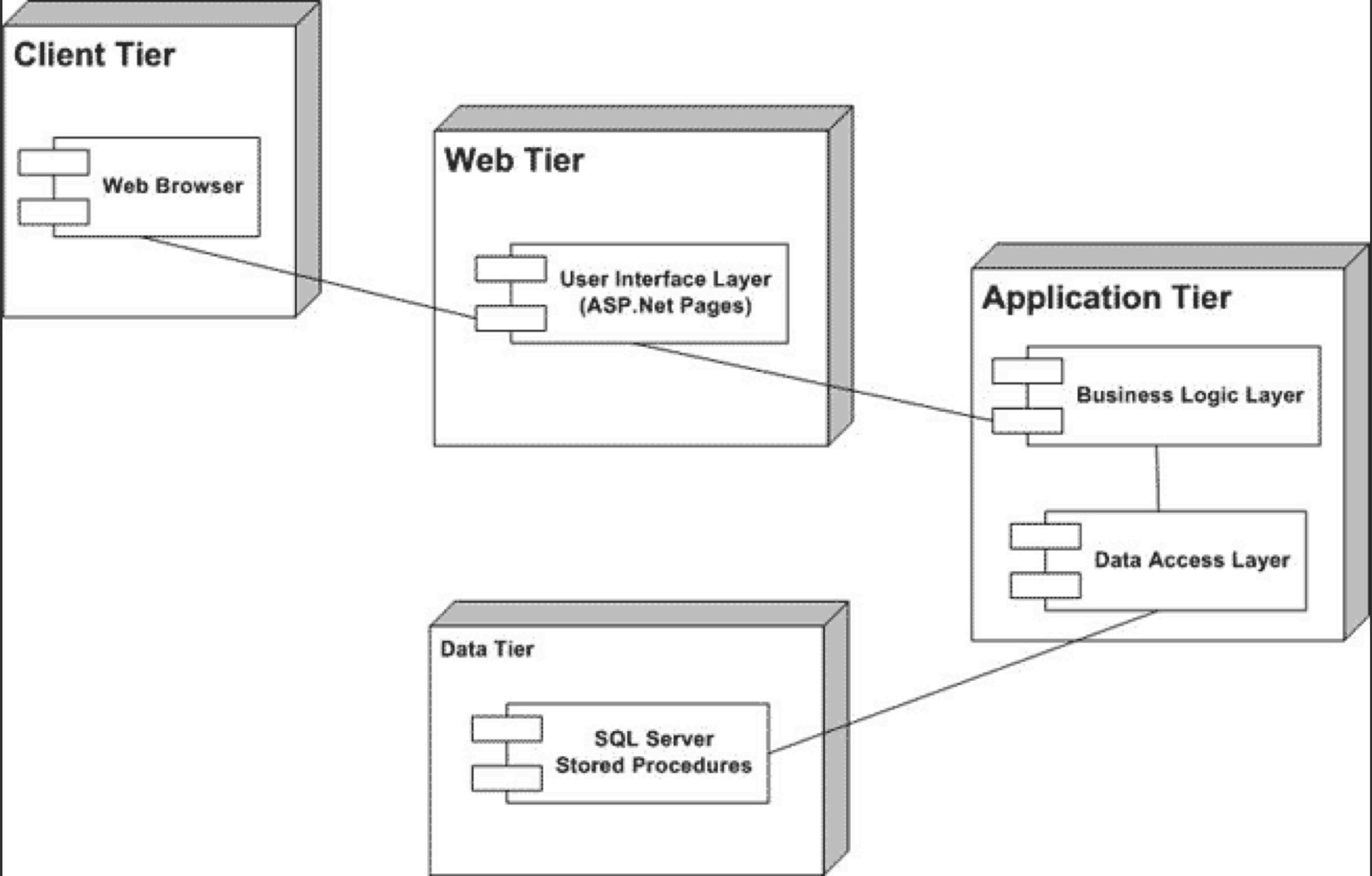
Collaboration "Rent Item", scenario "unavailable"



# Диагноз по Sequence → NFR

- Структура (внезапно)
- Ответственности компонентов
- Порядок операций
- Гранулярность операций
- Sync/async

# Шаблон №3: ВИДИМ качество за UML Deployment Diagram



**Диагноз по Sequence → NFR**

**– Самостоятельно**



# Точки зрения на систему для РМ

- План проекта с критическим путем
- План проверки гипотез по внешней и внутренней неопределенности

# Проверка внутренних рисков для РМ

- Прототипы
- Тесты
- Внутренняя экспертиза
- Внешняя экспертиза
- Запрос вендора

# Еще раз про фреймворк коммуникаций

- QA и PM в архитектурной группе
- Участие в решениях как можно раньше
- «Обоснуй» → через FR и NFR
- Накопление инженерной экспертизы в PoV
- Адаптированные PoV для других ролей
- *Архитектура – для всех!*

