
Software Engineering Conference Russia

14-15 ноября, 2019. Санкт-Петербург

# JSC

## новое поколение компонентной модели

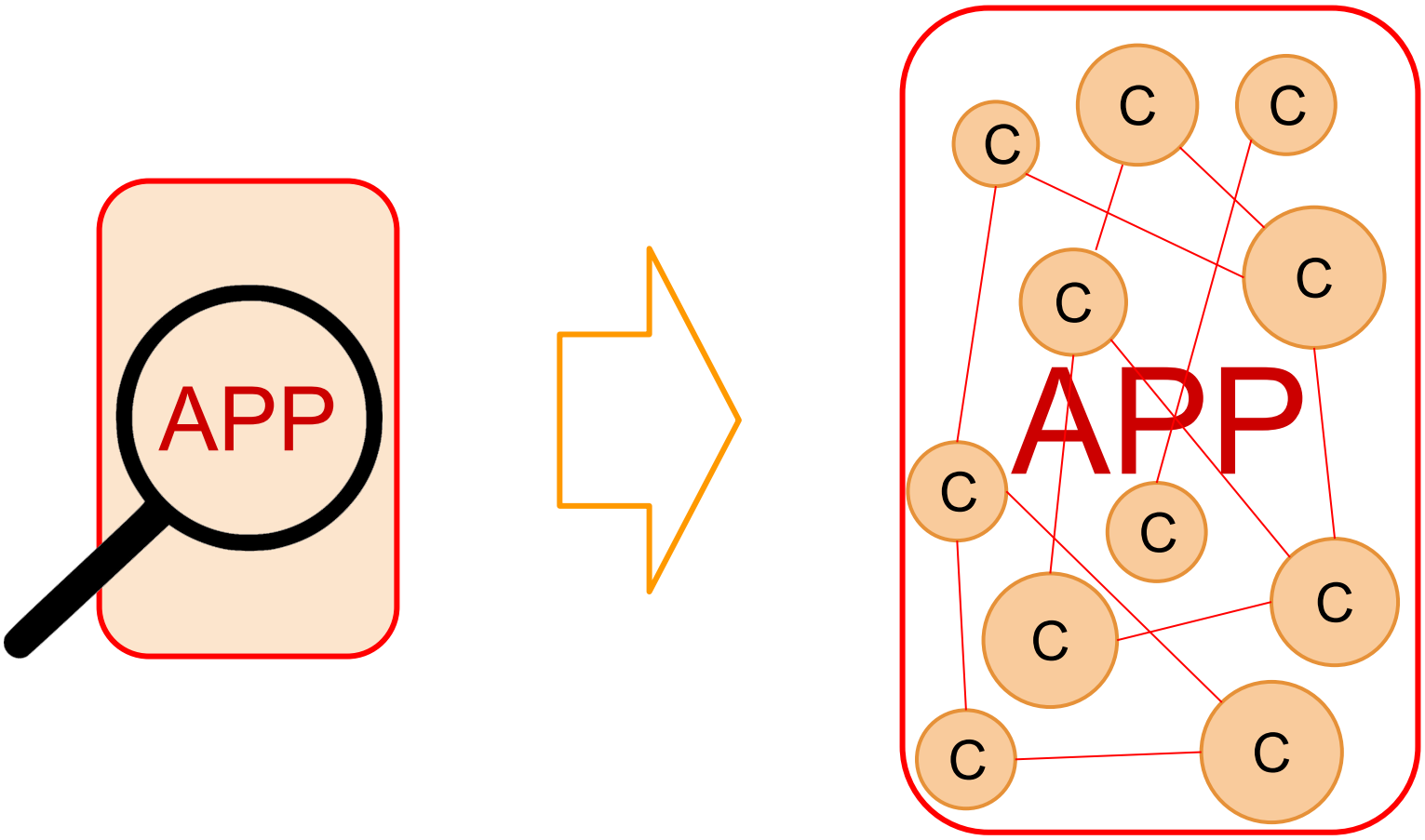
**Логинов Андрей**

**A7 Systems**

# ВВЕДЕНИЕ

# Классическое приложение






Изнутри приложение состоит из компонент

# Что такое компонент?



**Библиотека**

**Класс**

**Структура**

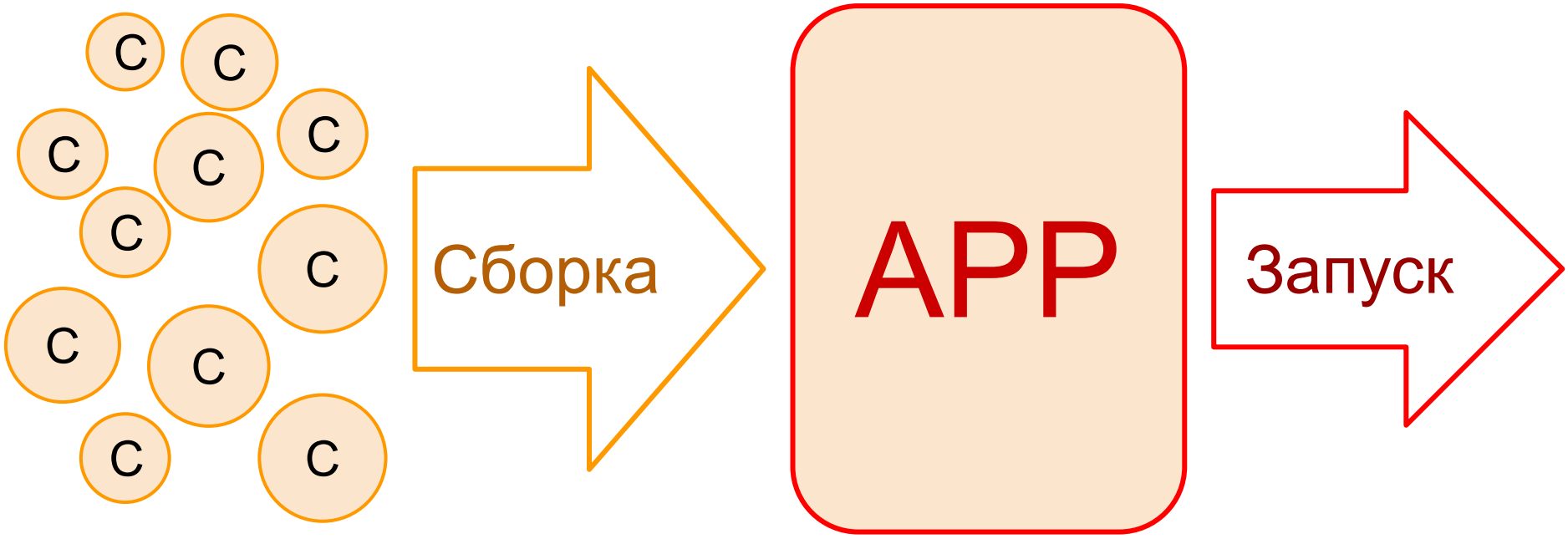
**Пакет**

**Функция**

Зависит от языка, среды, стиля программирования

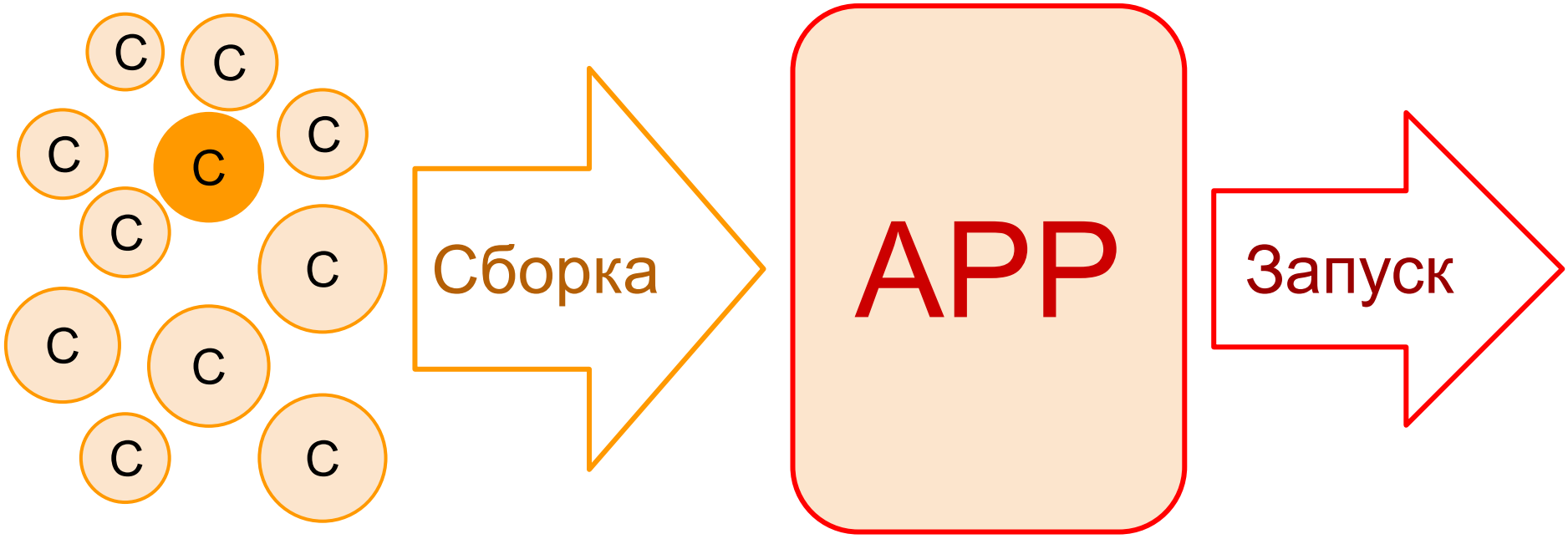
# Как это работает






Создаем компоненты, собираем, запускаем

# Изменения?




Изменяем компоненты, собираем, перезапускаем

# Борьба с остановкой



**Библиотека**

**Кластеризация**

**Скрипт**

Не универсальное средство и сложно

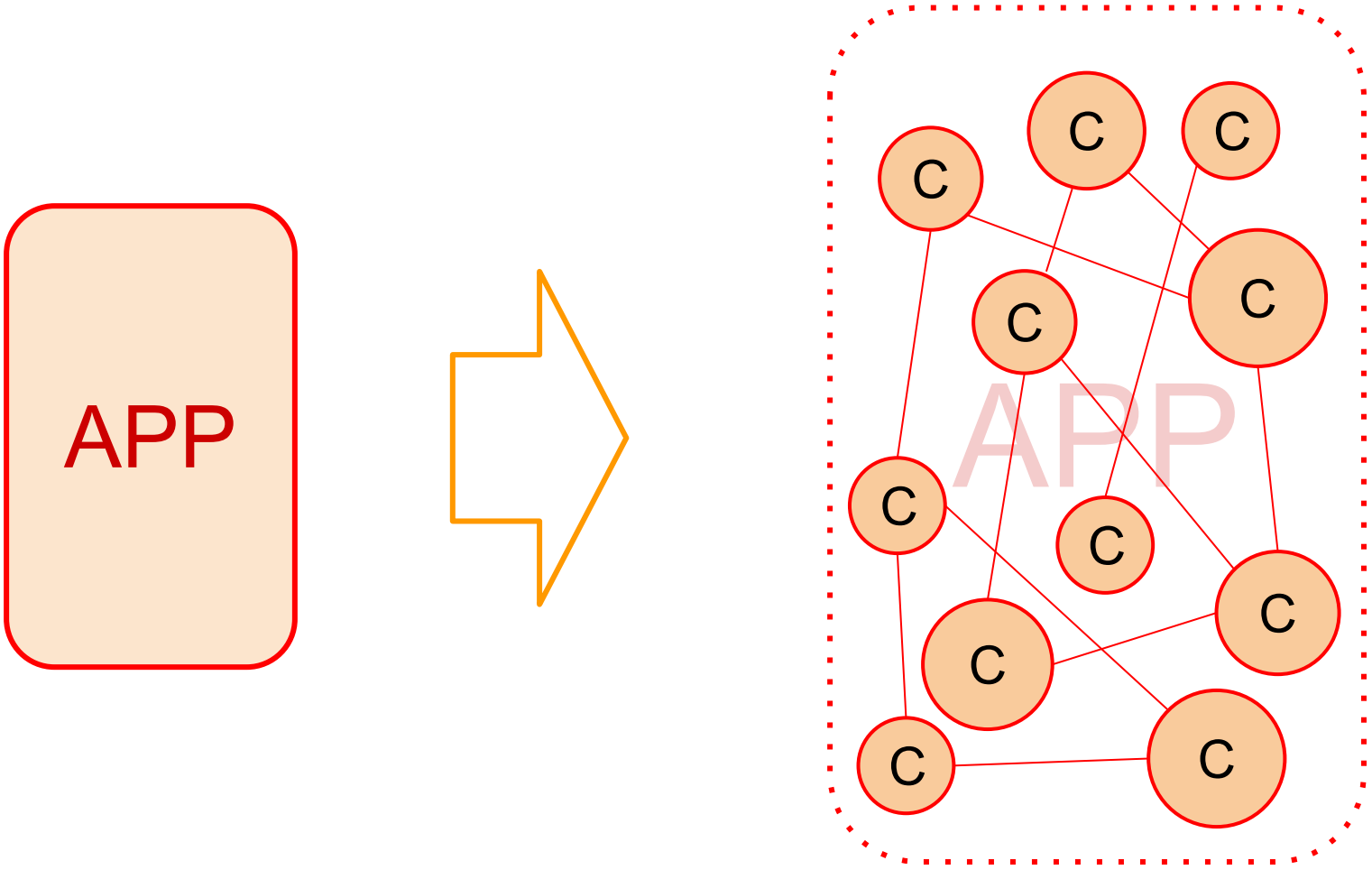
# Заголовок



JSC

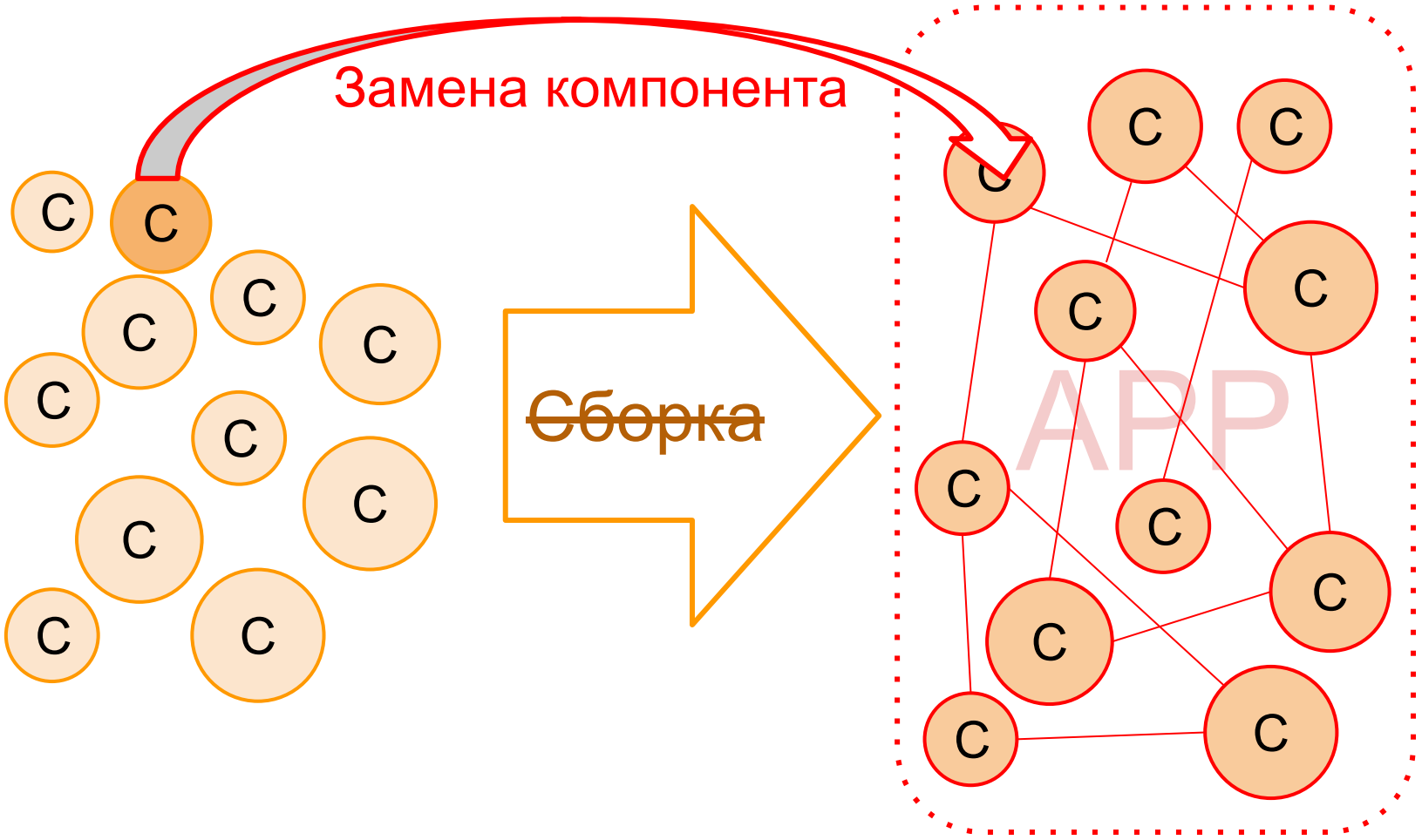
# Изменим подход





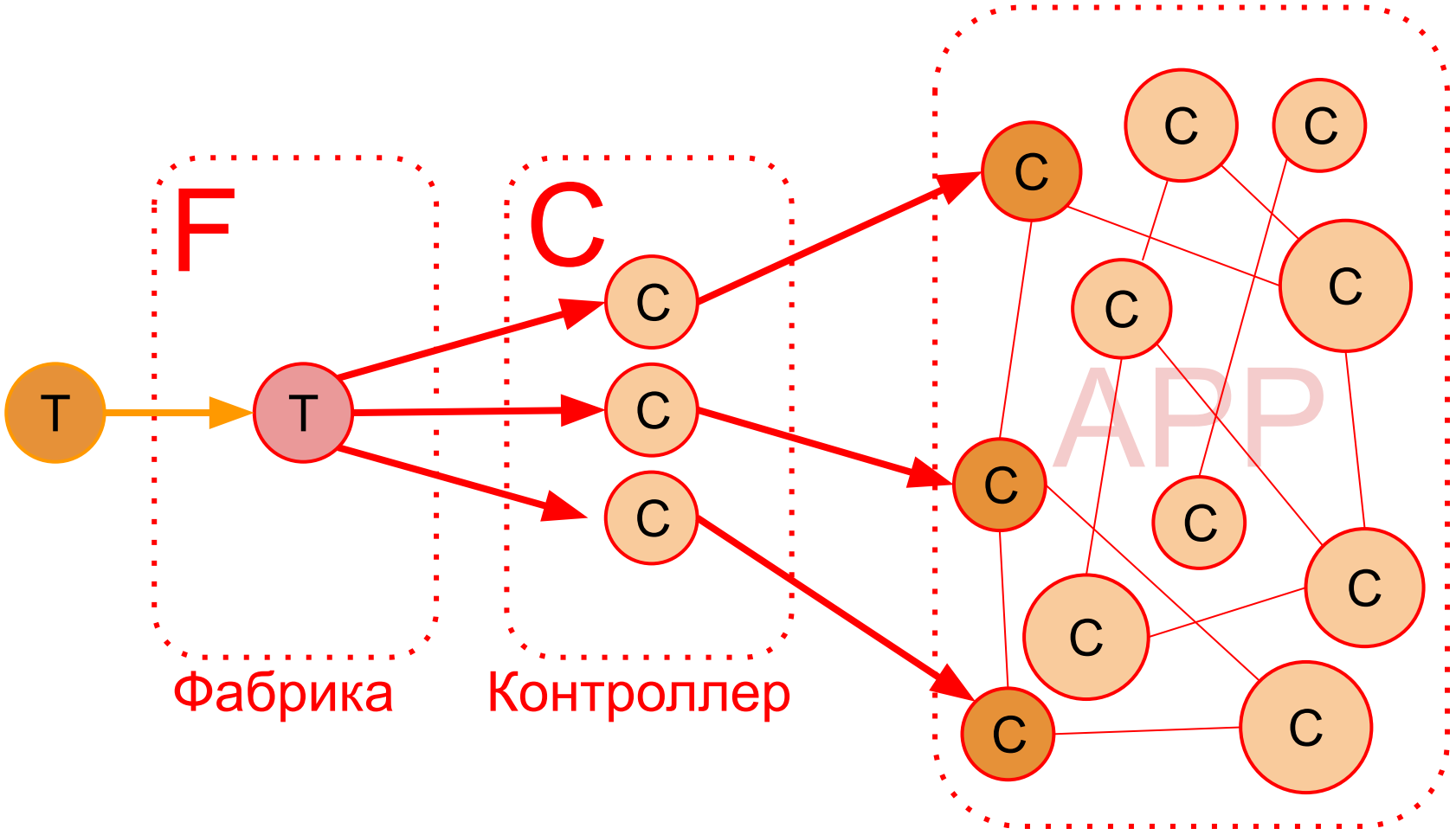

Уберем границы приложения

# Изменения?




~~Пересобираем, перезапускаем~~

# На практике немного сложнее




Мы изменяем тип, на основе которого меняются экземпляры

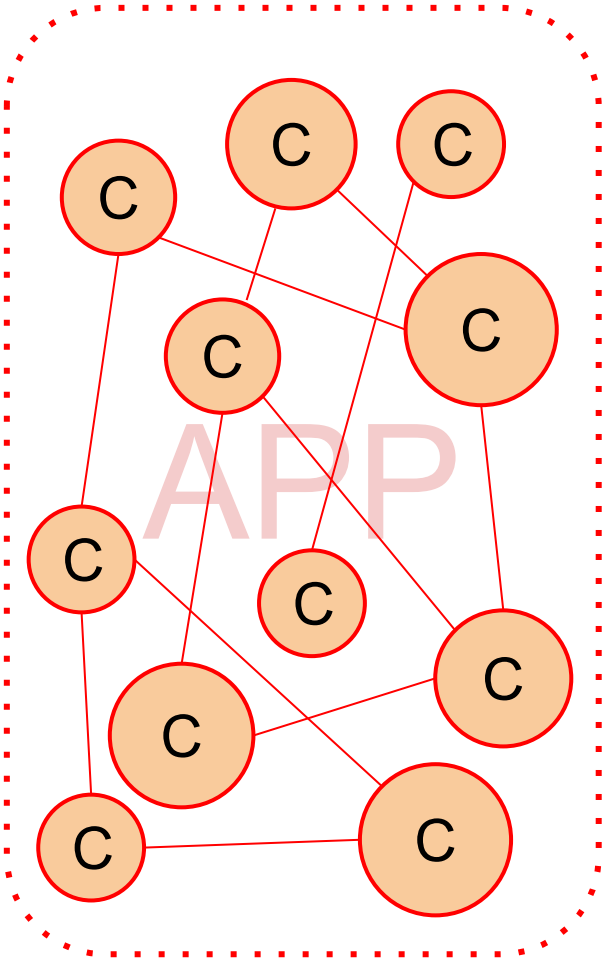
# Собственно это фундамент JSC



JSC

F фабрика

C контроллер




Возможность произвольно изменять приложение в процессе работы

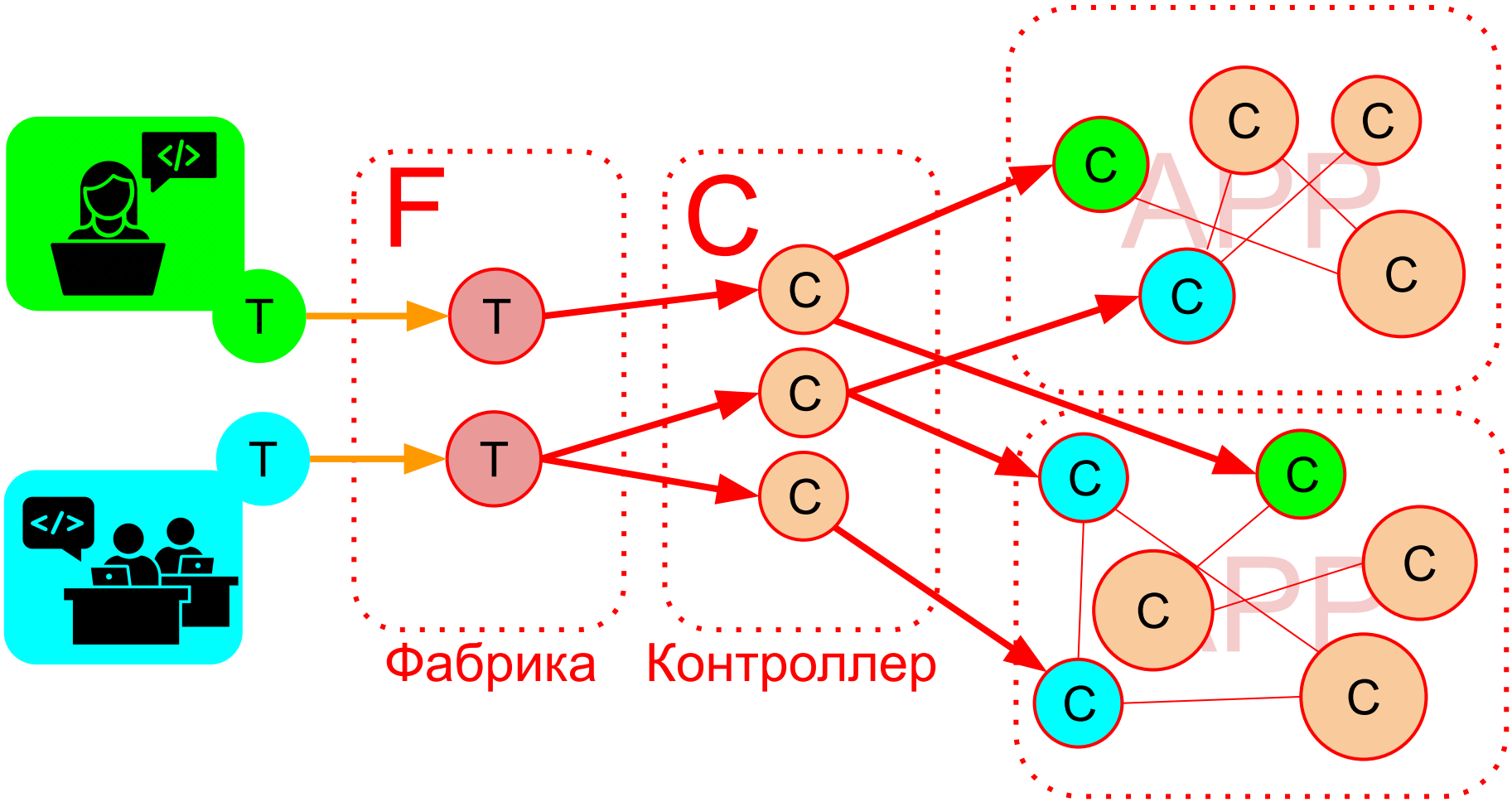
# Но JSC это не только код



**Компоненты**

**это результат работы**

**разработчиков**




Объединение усилий разработчиков

# Должны выполняться принципы



**Возможность замены и улучшения**

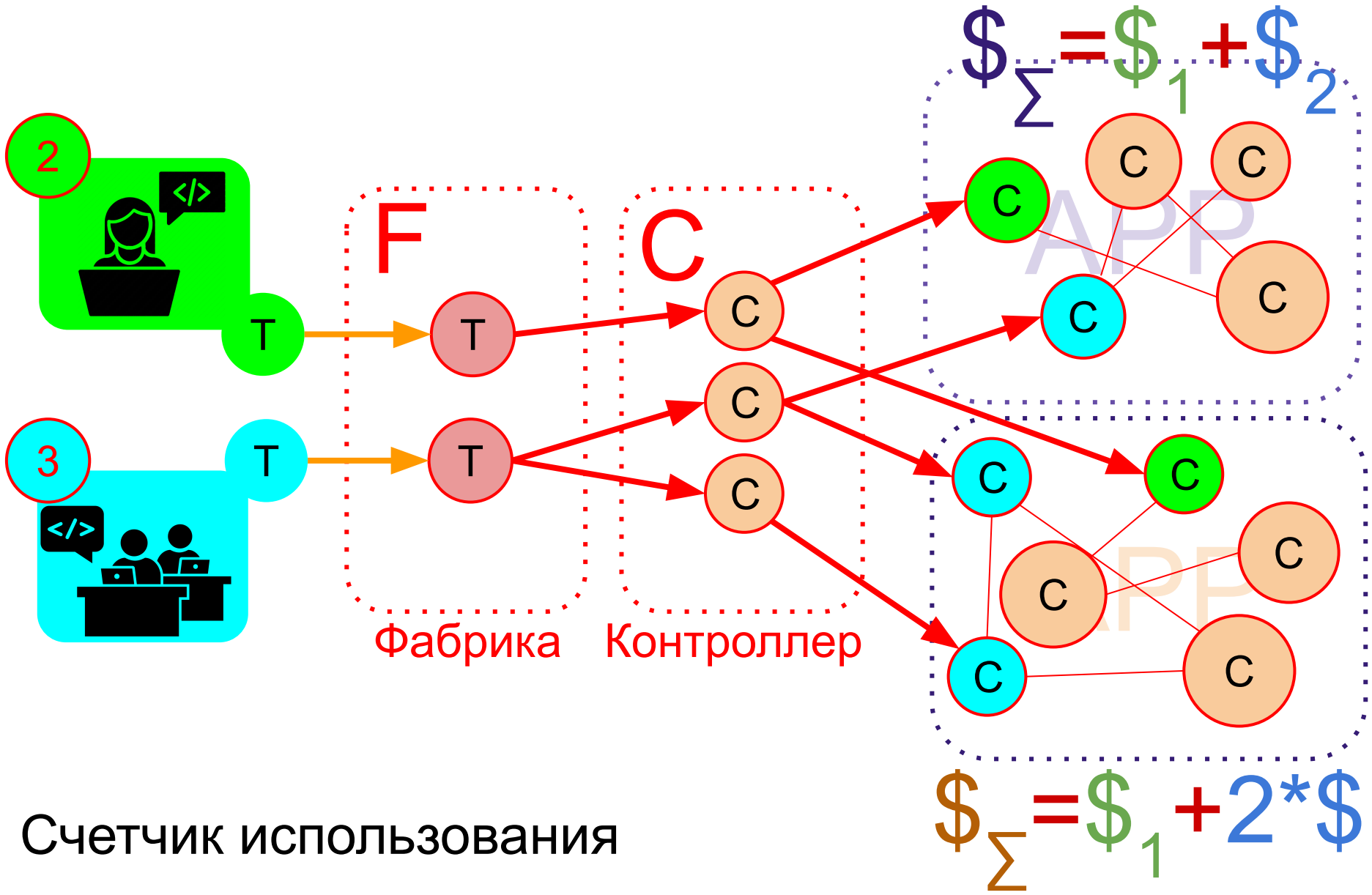
**Безопасность использования**

**Соблюдение лицензий**

Одновременно!

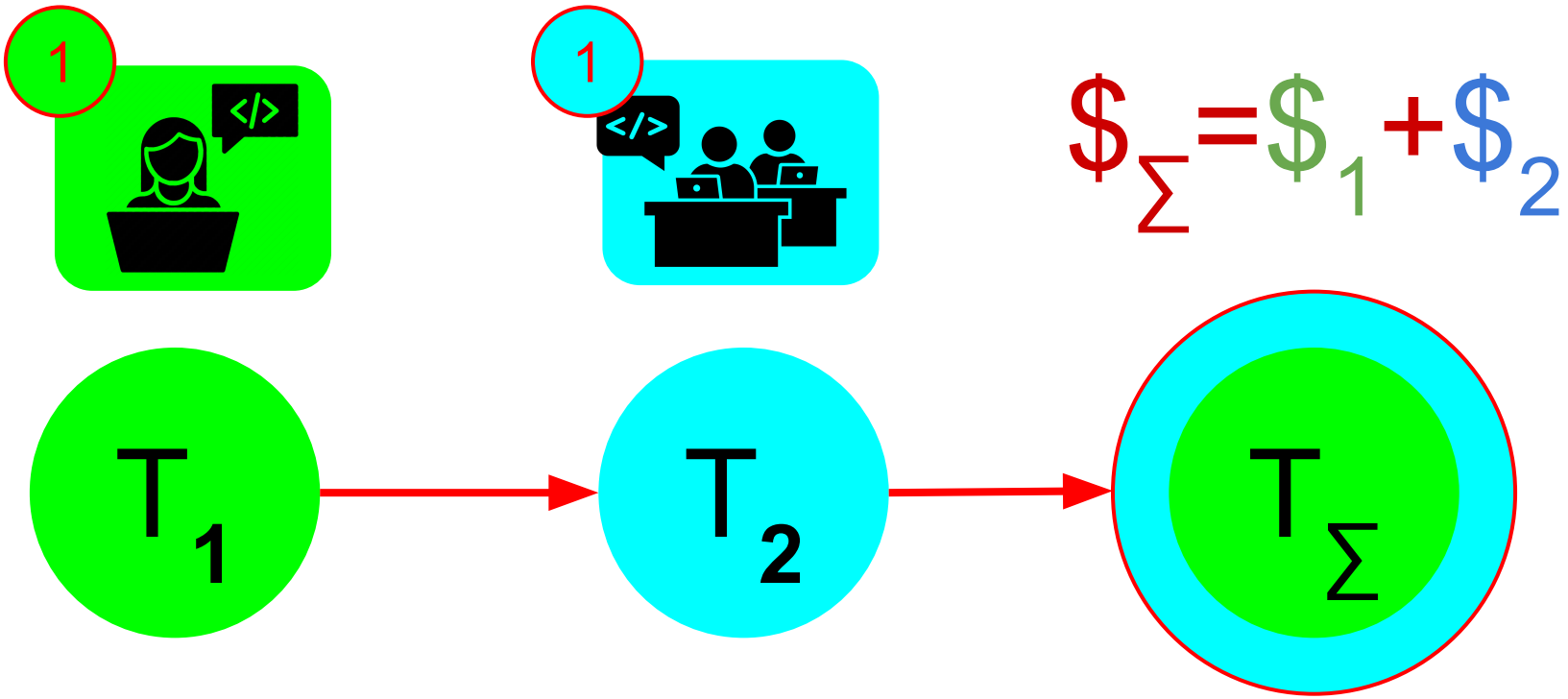
# ЛИЦЕНЗИРОВАНИЕ

# Монетизация




Счетчик использования

# Наследование




Добавленная стоимость

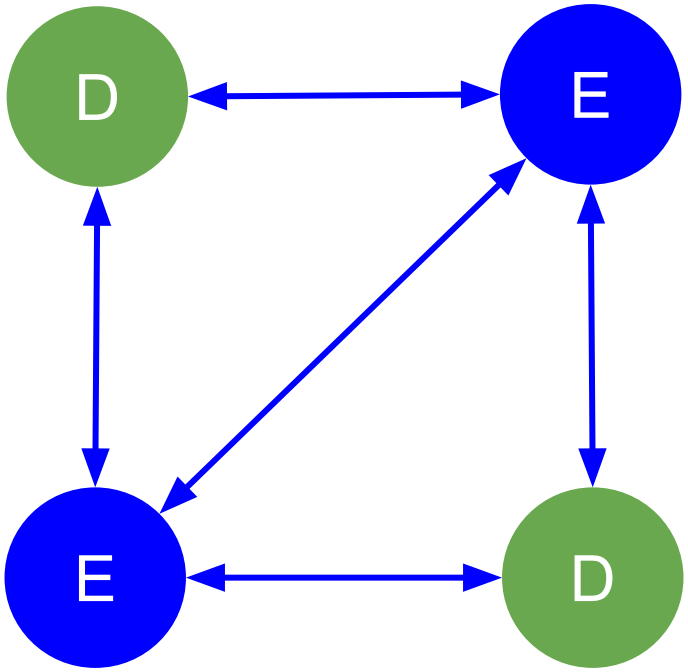
# Лицензирование



## Контроль использования компонент

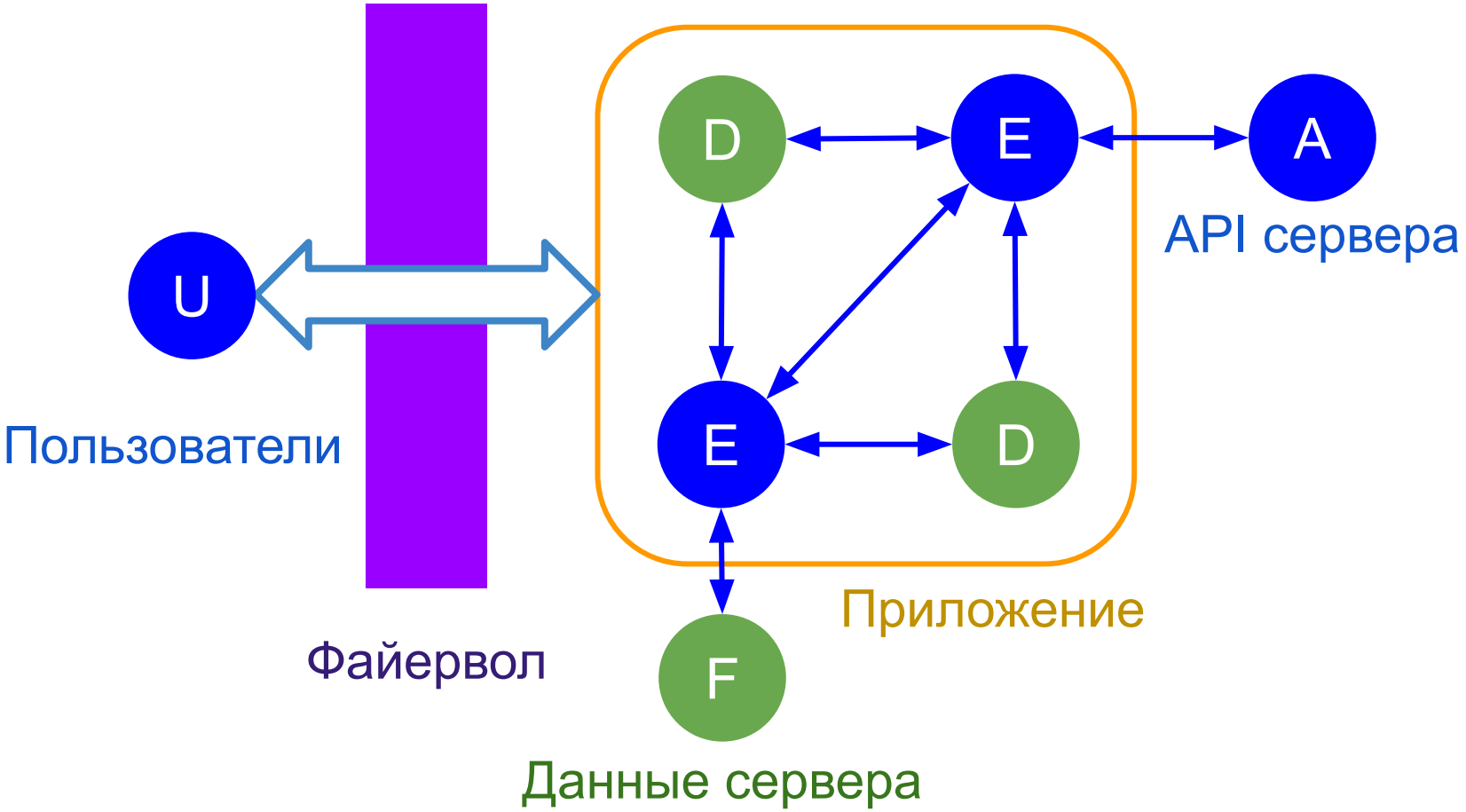
# БЕЗОПАСНОСТЬ

# Программа – код и данные




Код модифицирует данные

# Мы ***доверяем*** коду




А потом удивляемся

# Доступ для кода

Уровень доступа пользователя

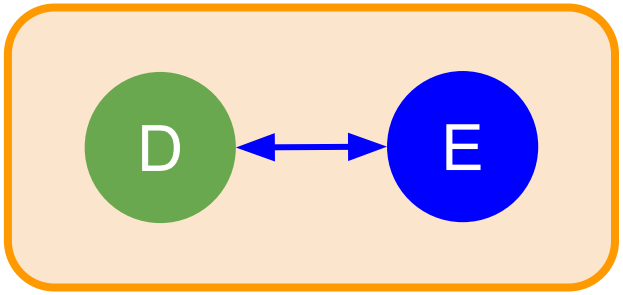
**+ Уровень доступа компонента**

Двойной контроль позволяет безопасно работать с данными

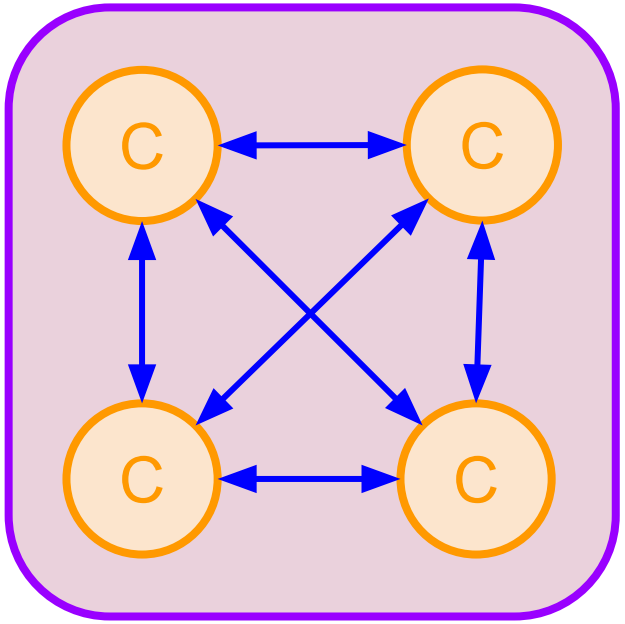
# Код компонента это черный ящик



Компонент




Пакет



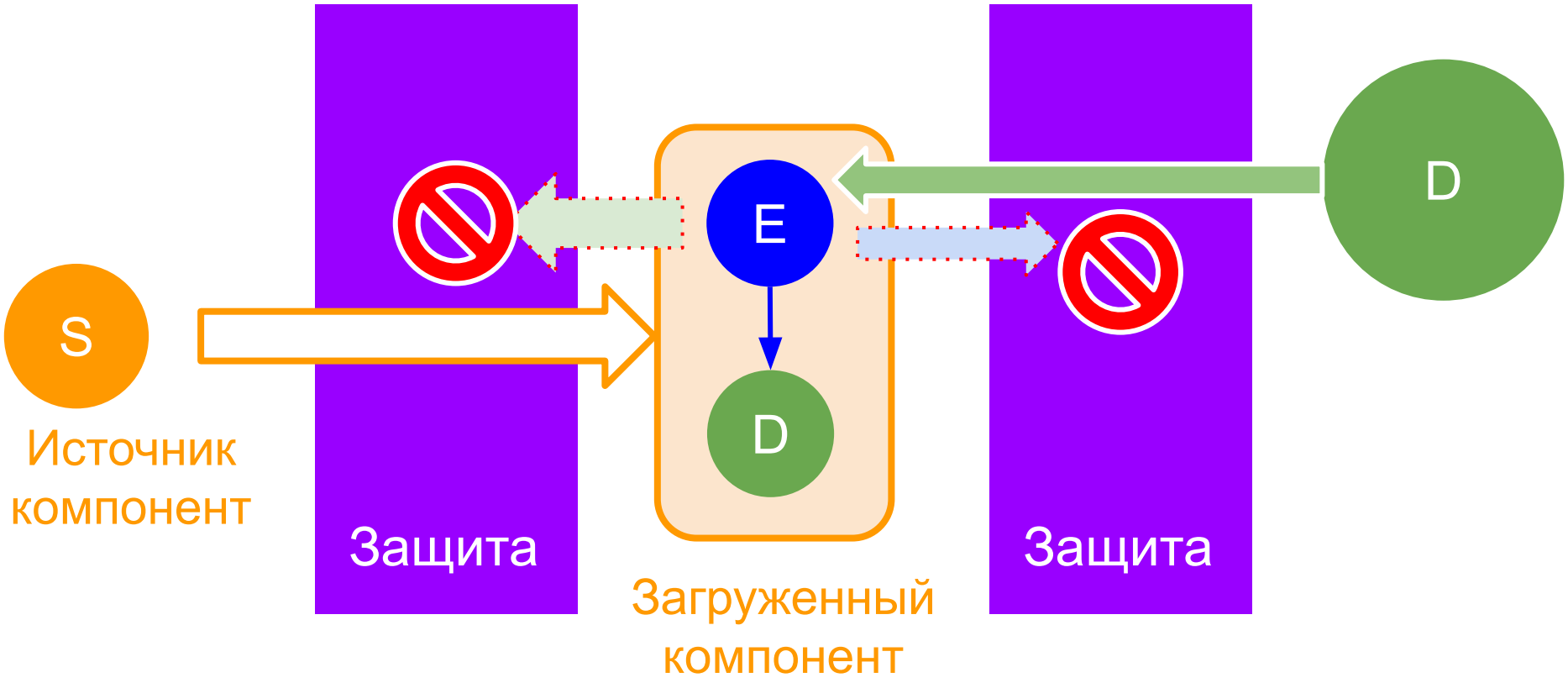

Компонент это код и его данные

Пакет содержит код и данные своих компонентов

В большинстве случаев нет смысла контролировать доступ внутри пакета или компонента

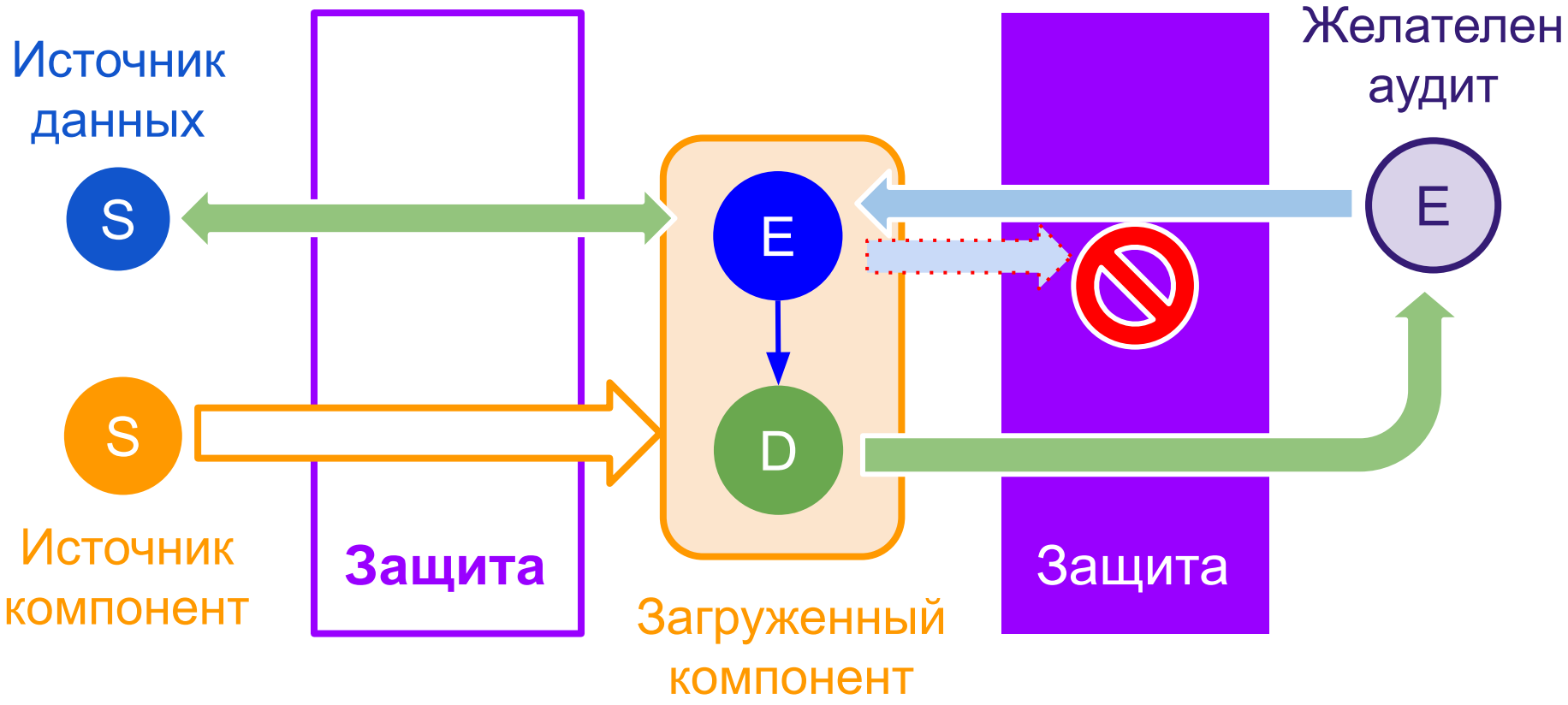
# Сценарий для аналитики





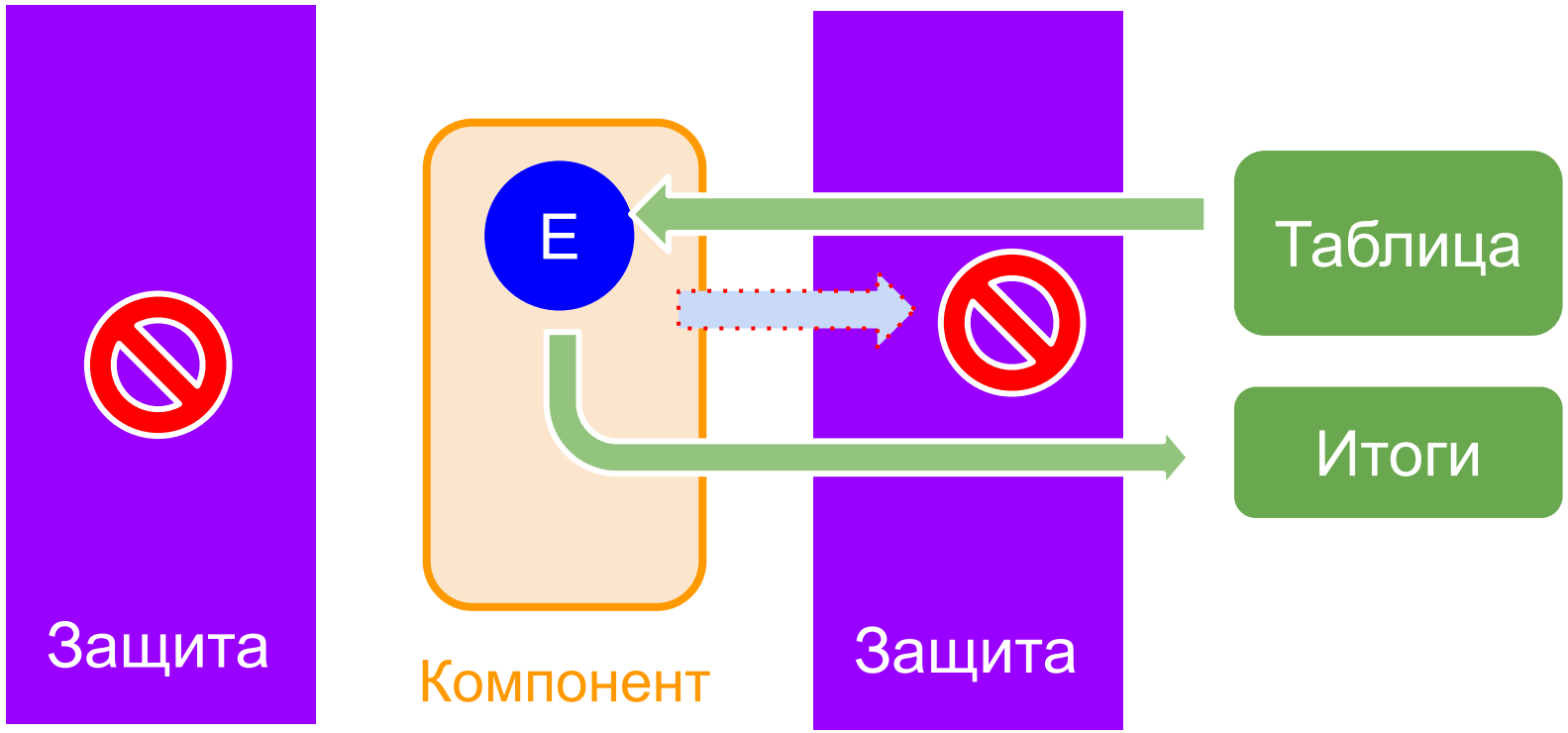

Запрет менять и передавать данные, кроме как внутри компонента/пакета

# Сценарий для сервиса




Запрет менять данные, кроме как внутри компонента/пакета

# Пример сценария



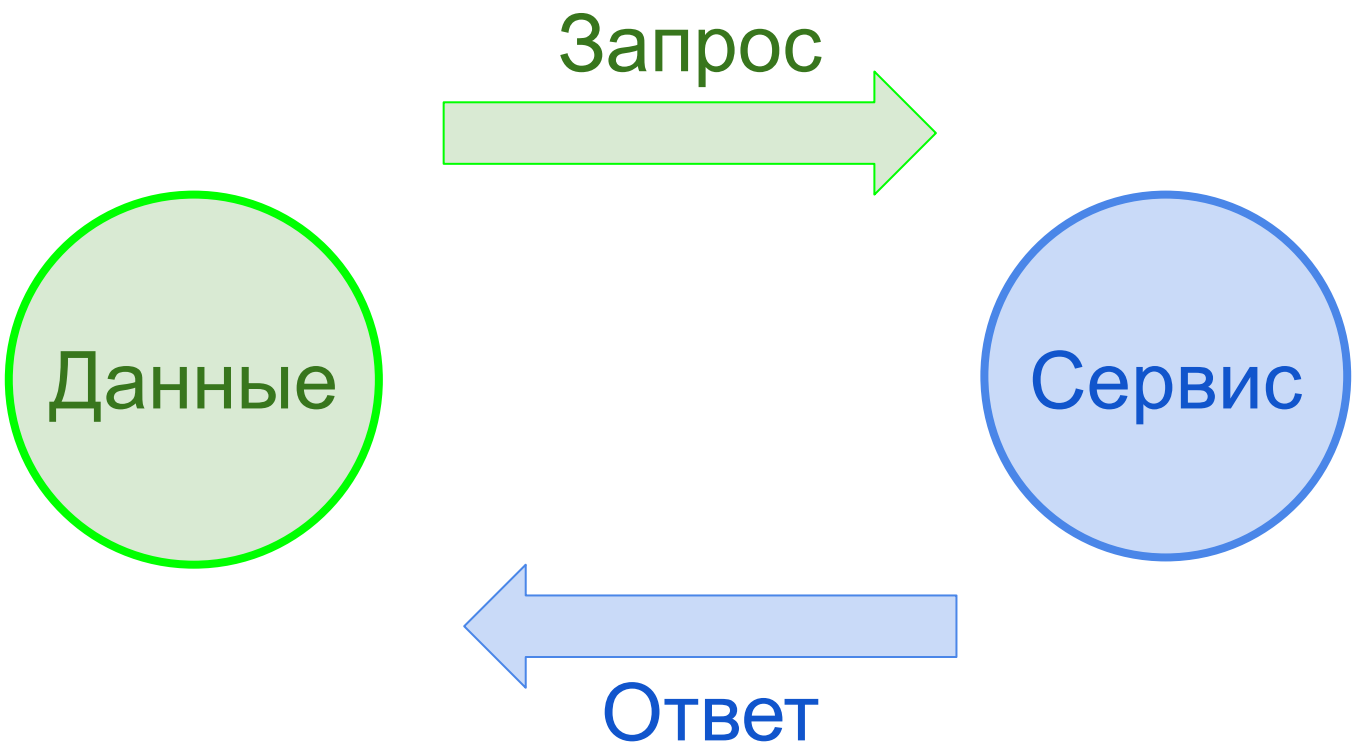

Разрешение на конкретные изменения

# Заголовок

# БИГ ДАТА

# Как мы работаем с данными

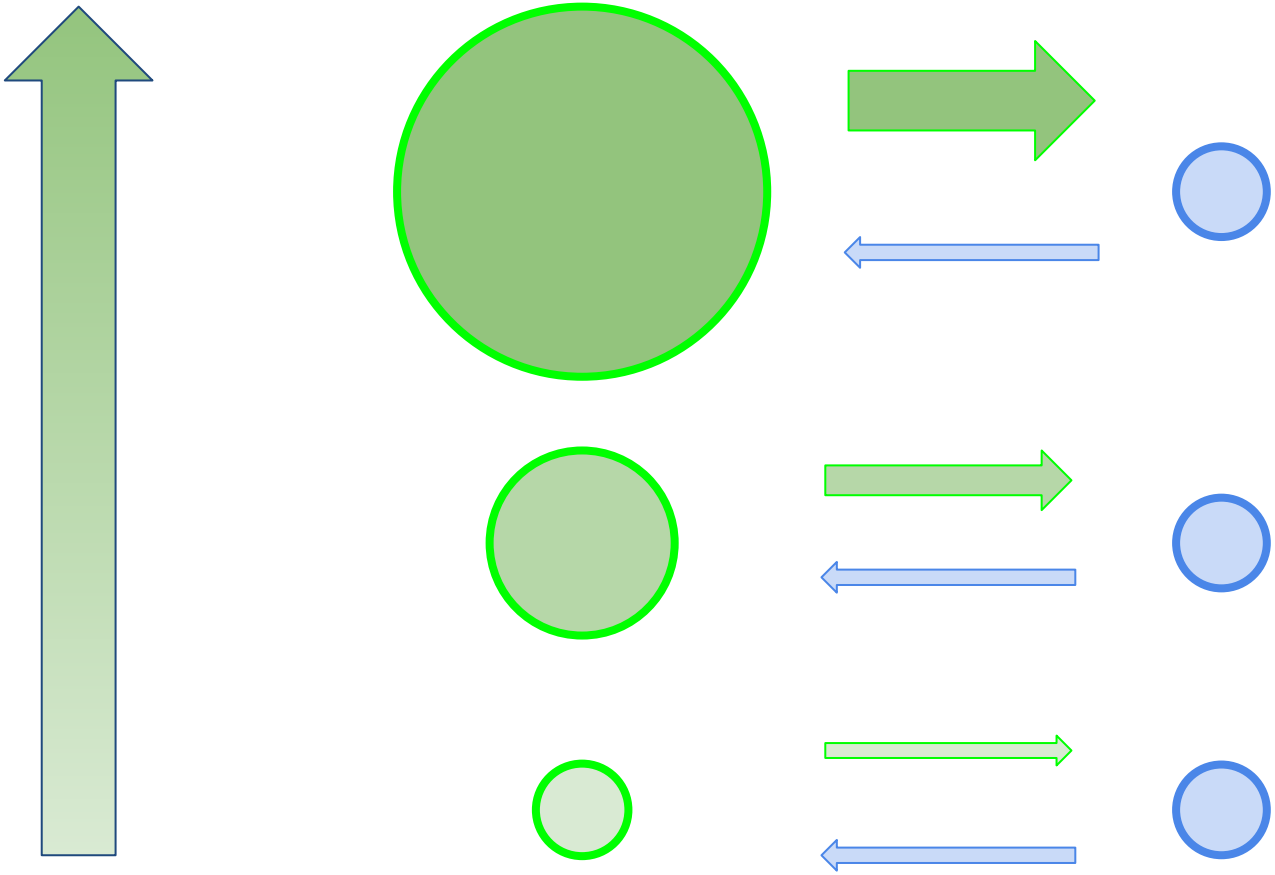
Правда данные могут украсть




Мы отправляем данные на обработку и получаем результат

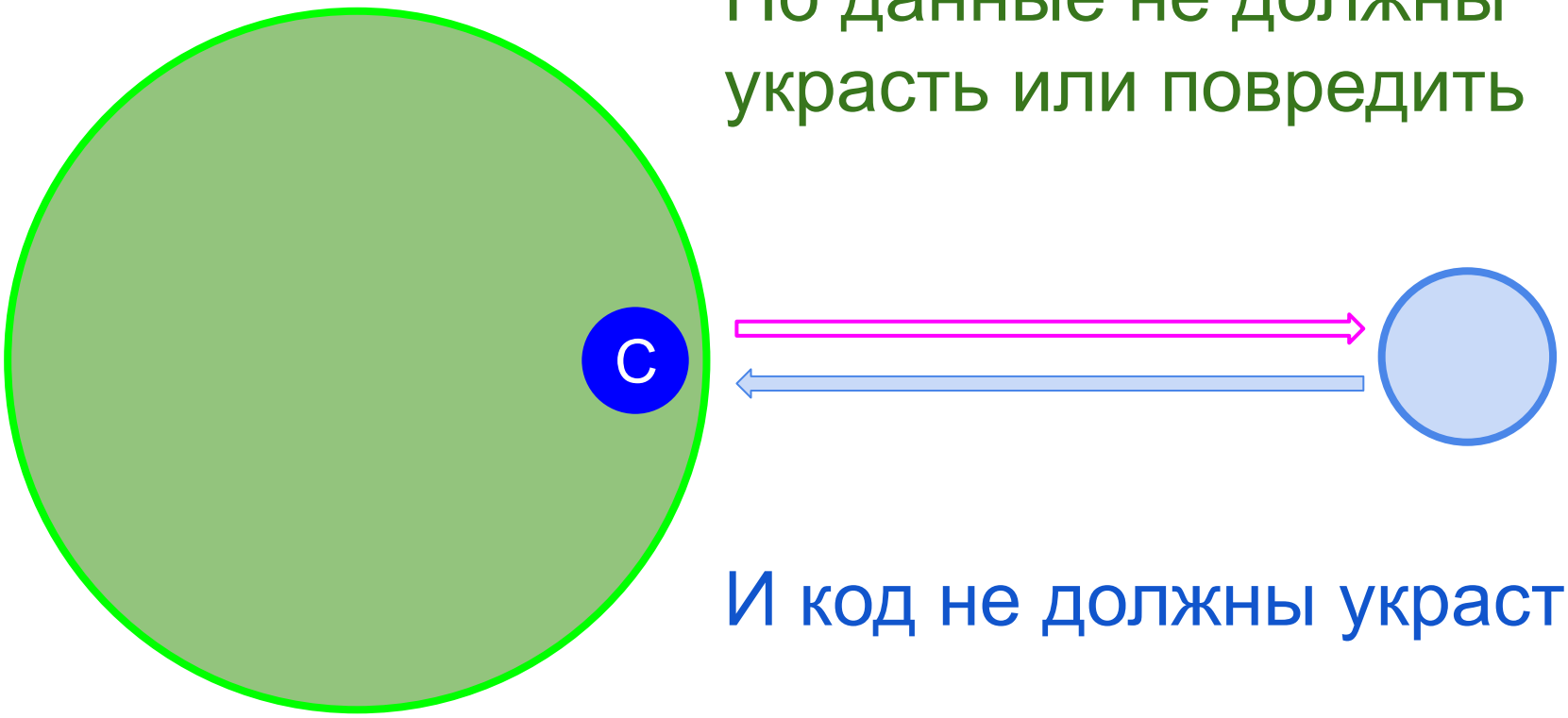
# Потом данных становилось больше





Количество данных растет, и схема перестает быть адекватной

# Логичнее загрузить код в данные




Но данные не должны украсть или повредить

И код не должны украсть

# JSC и позволяет это делать

Уровень доступа пользователя

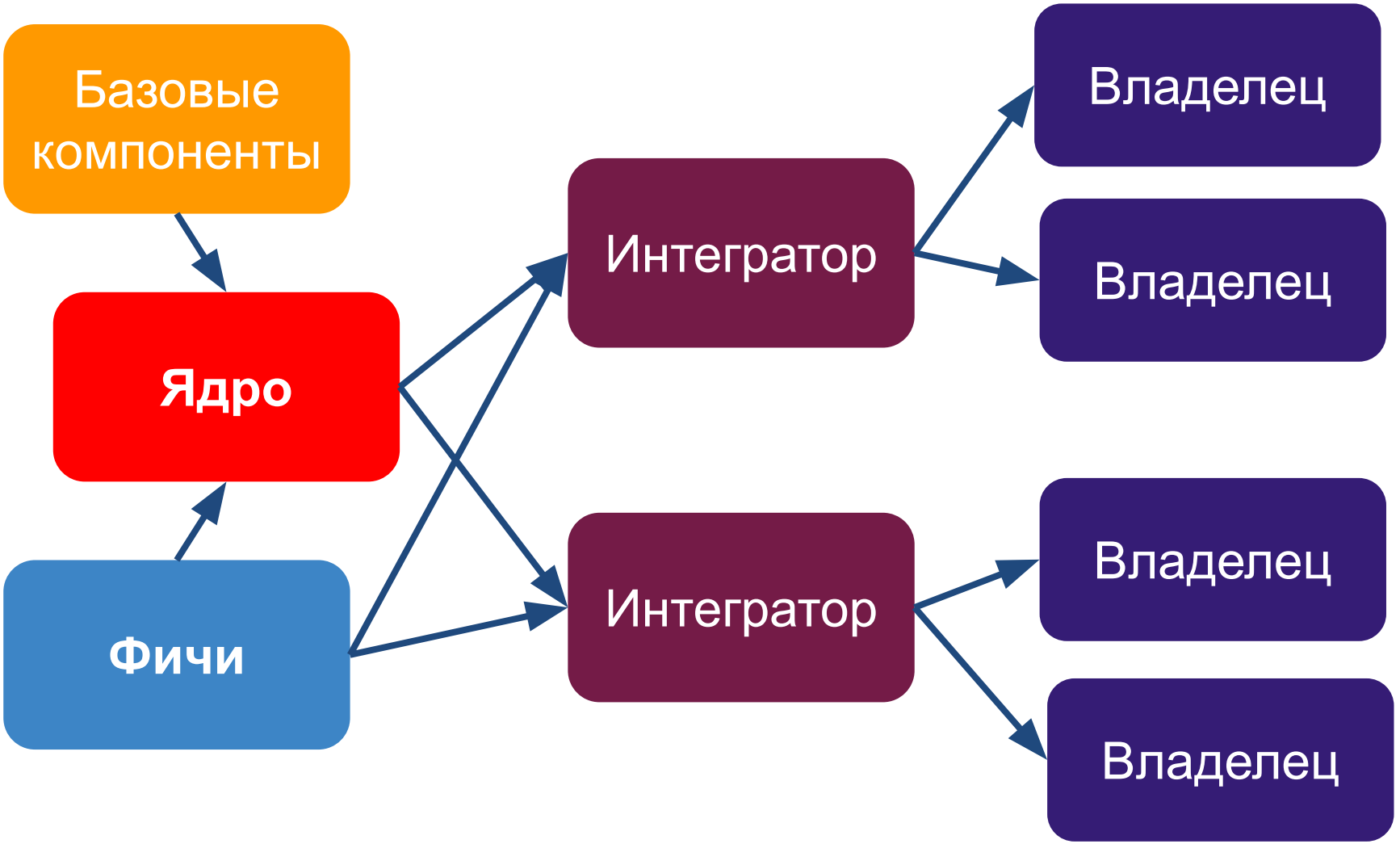
\+ **Уровень доступа компонента**

\+ **Лицензионная защита**

Условия для корректной загрузки компонент

# ВЕРСИОННОСТЬ

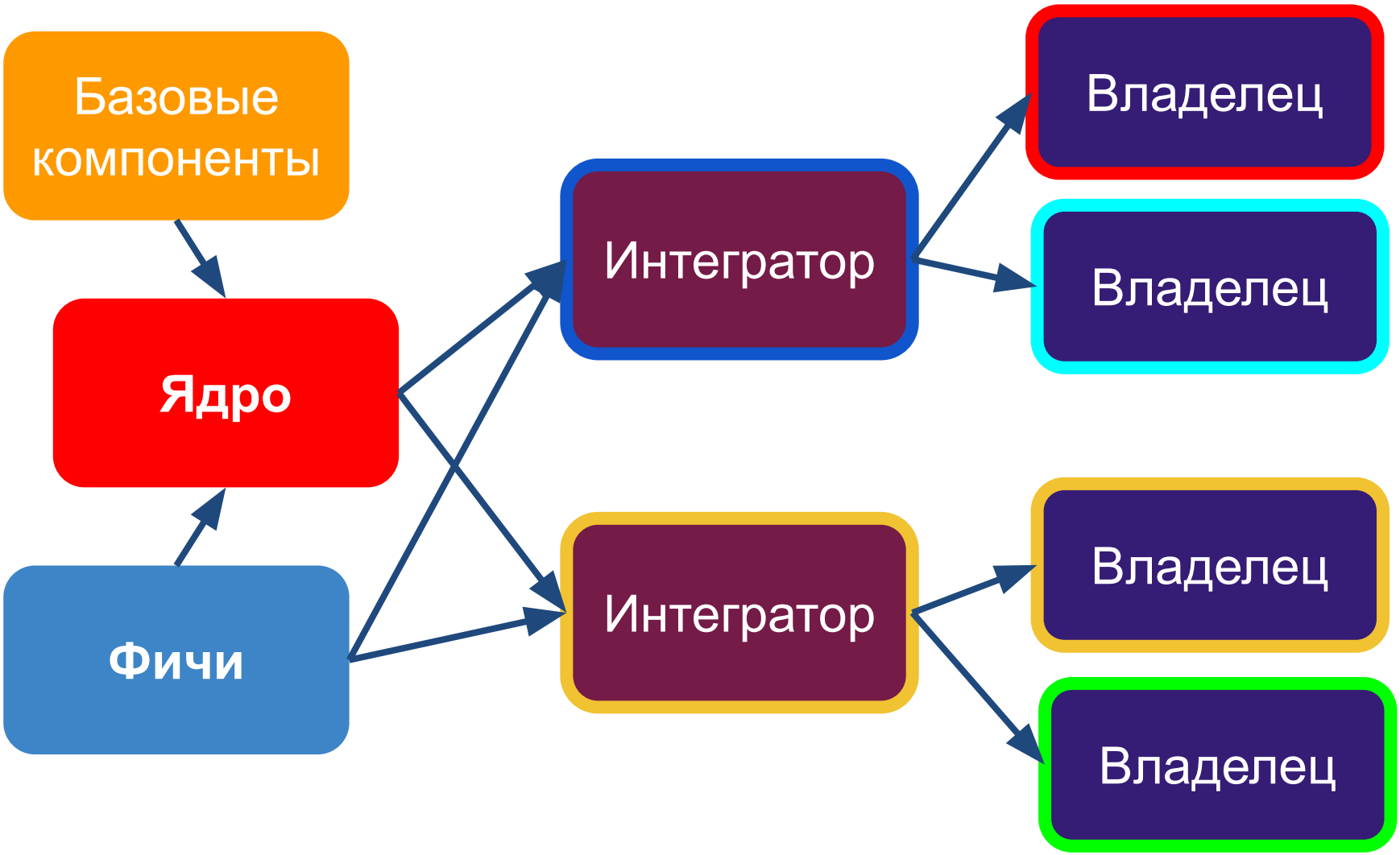
# Немного практики



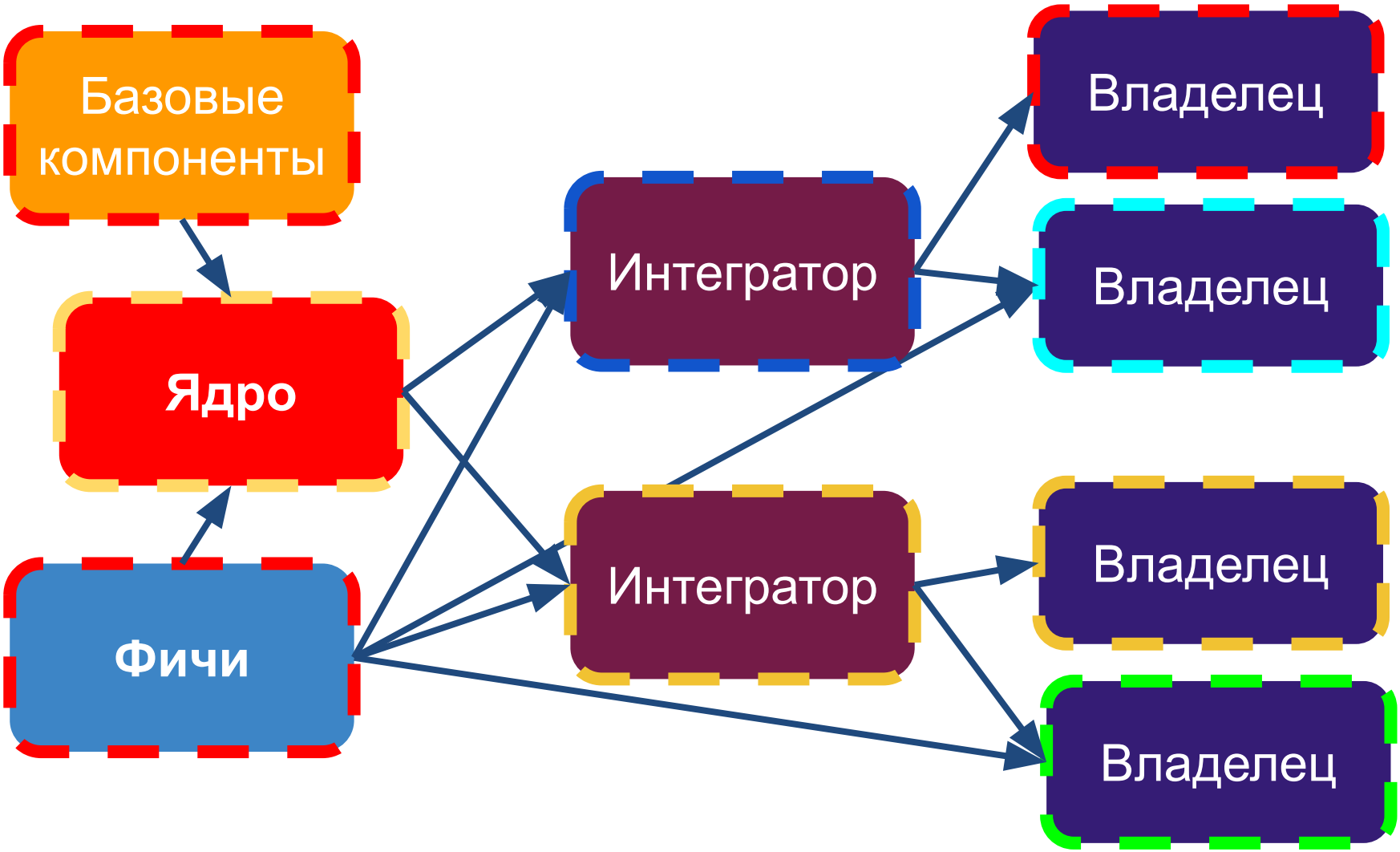

Объединение усилий разработчиков

# Каждый на своем этапе улучшил





# Мир постоянно меняется




Так выглядит ад улучшений

# Обновление в JSC



механизм отслеживания изменений

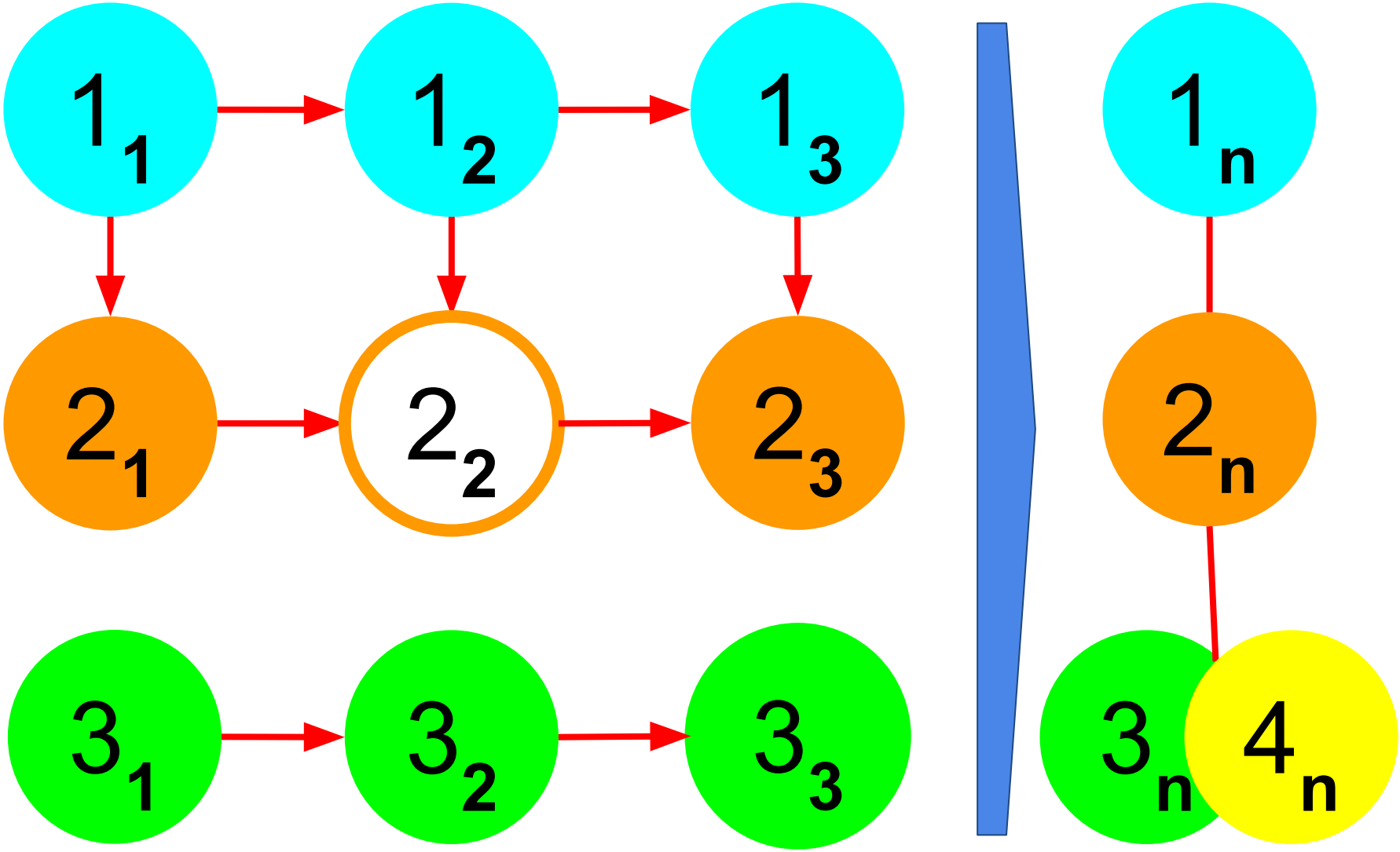
обновление потомков

и

использующих конфигураций

# Используется любая версия 1...N





# Версионность JSC



использование альтернативных компонент или версий

# Лицензирование



Контроль использования компонент

**+ Гарантия доступности компонент**

# Заголовок

# ПРОБЛЕМЫ

# Проблемы



Закрытость байт-машин и компиляторов

Байт машина не дает 100% лицензионной защиты

Ограничения лицензионного контроля для Web и QML

# ПЕРСПЕКТИВЫ

# Кто будет развивать JSC



Google

Microsoft

Apple

Oracle

Unity3D

# Заголовок

**СМЫСЛ**

# Зачем все это?



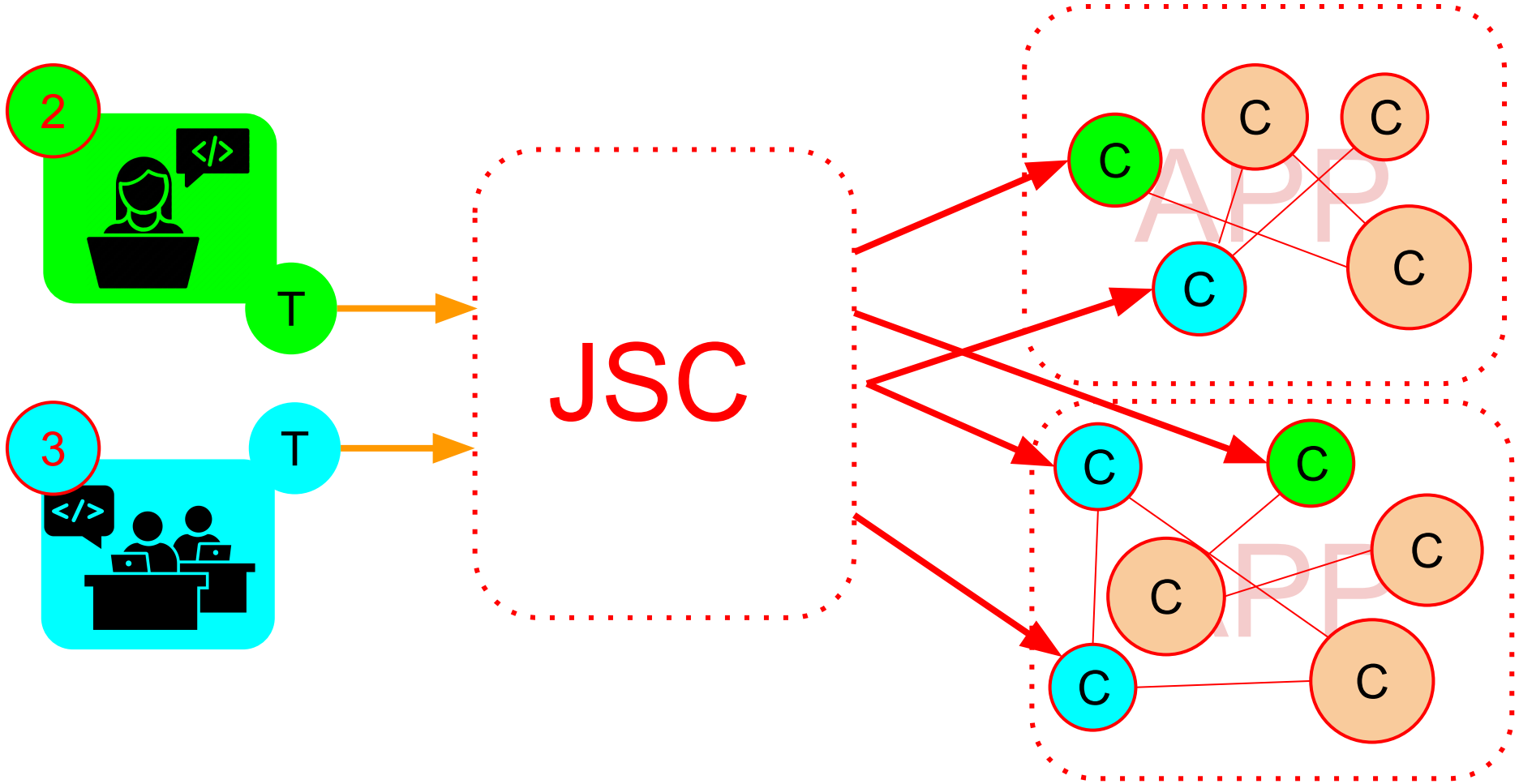
**Скорость разработки**

**Адаптивность**

**Качество решений**

# JSC – новые экономические отношения






Более совершенная модель производства

# Спасибо за внимание!

# JSC


## Андрей Логинов

**A7 Systems**

and-log@a7systems.org

www.a7systems.org

www.jooa.org