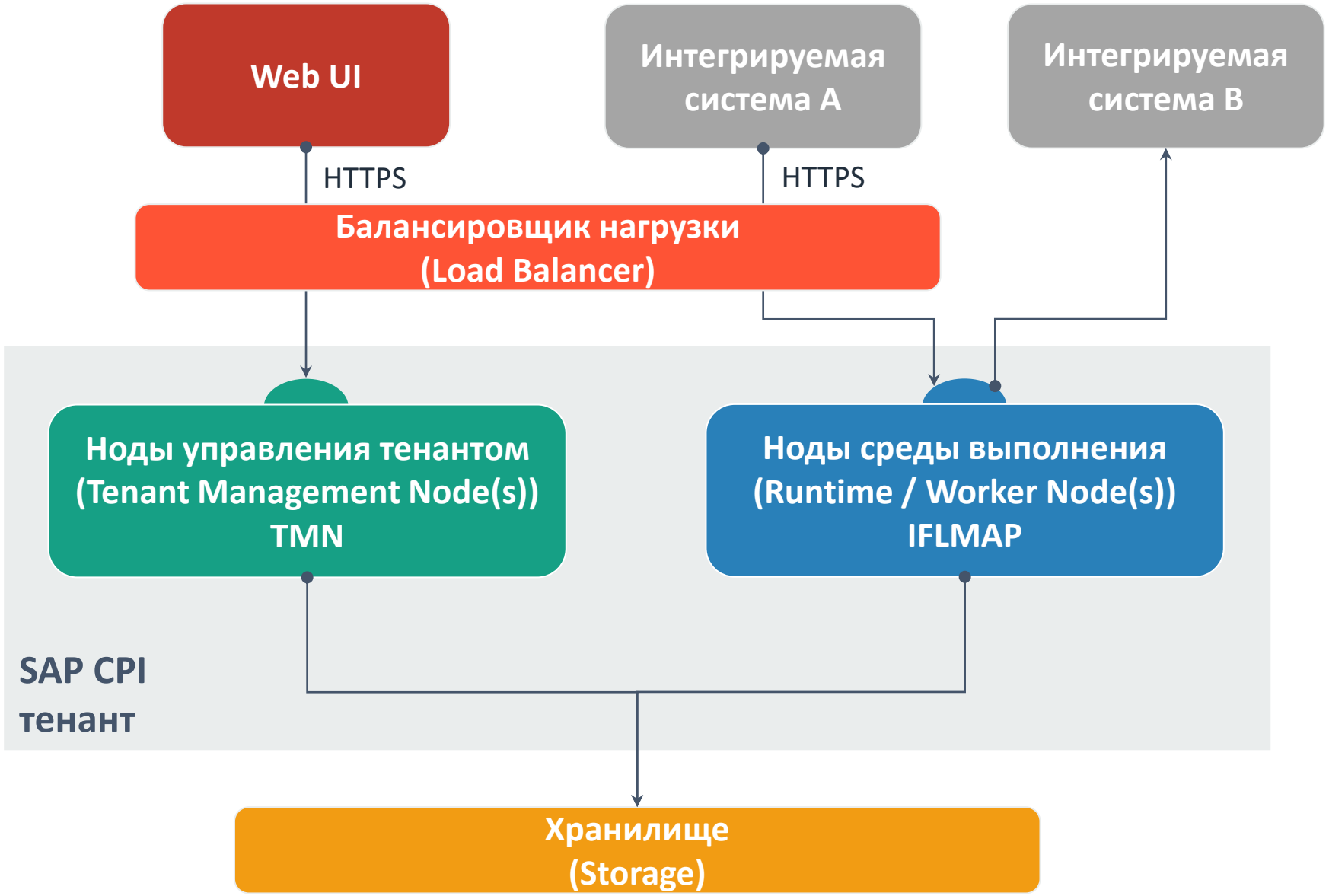
Архитектура среды выполнения SAP Cloud Platform Integration (Cloud Foundry)

Вадим Климов

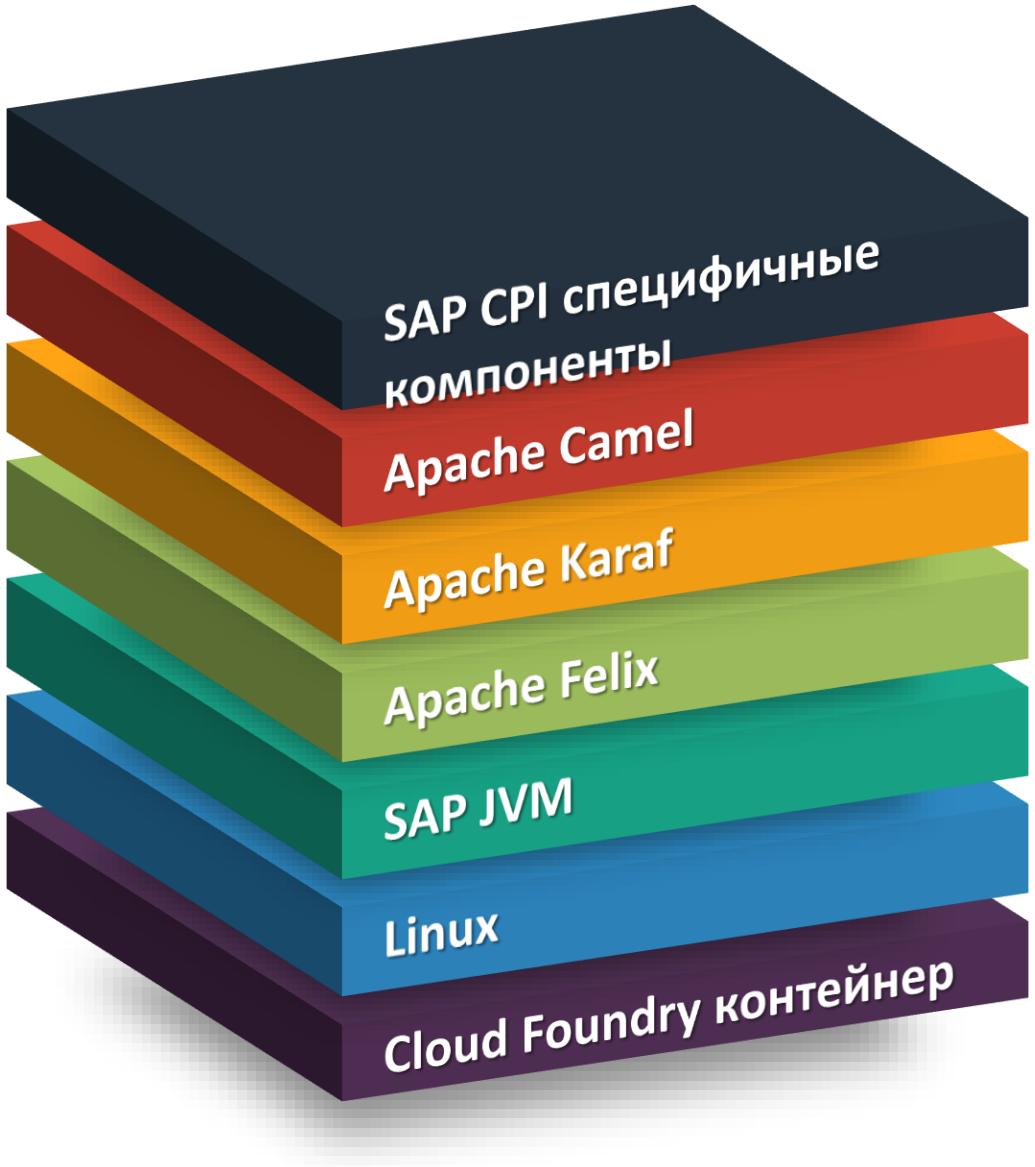
4 сентября, 2020

Архитектура SAP CPI тенанта



Нода среды выполнения SAP CPI (SAP CPI runtime node)

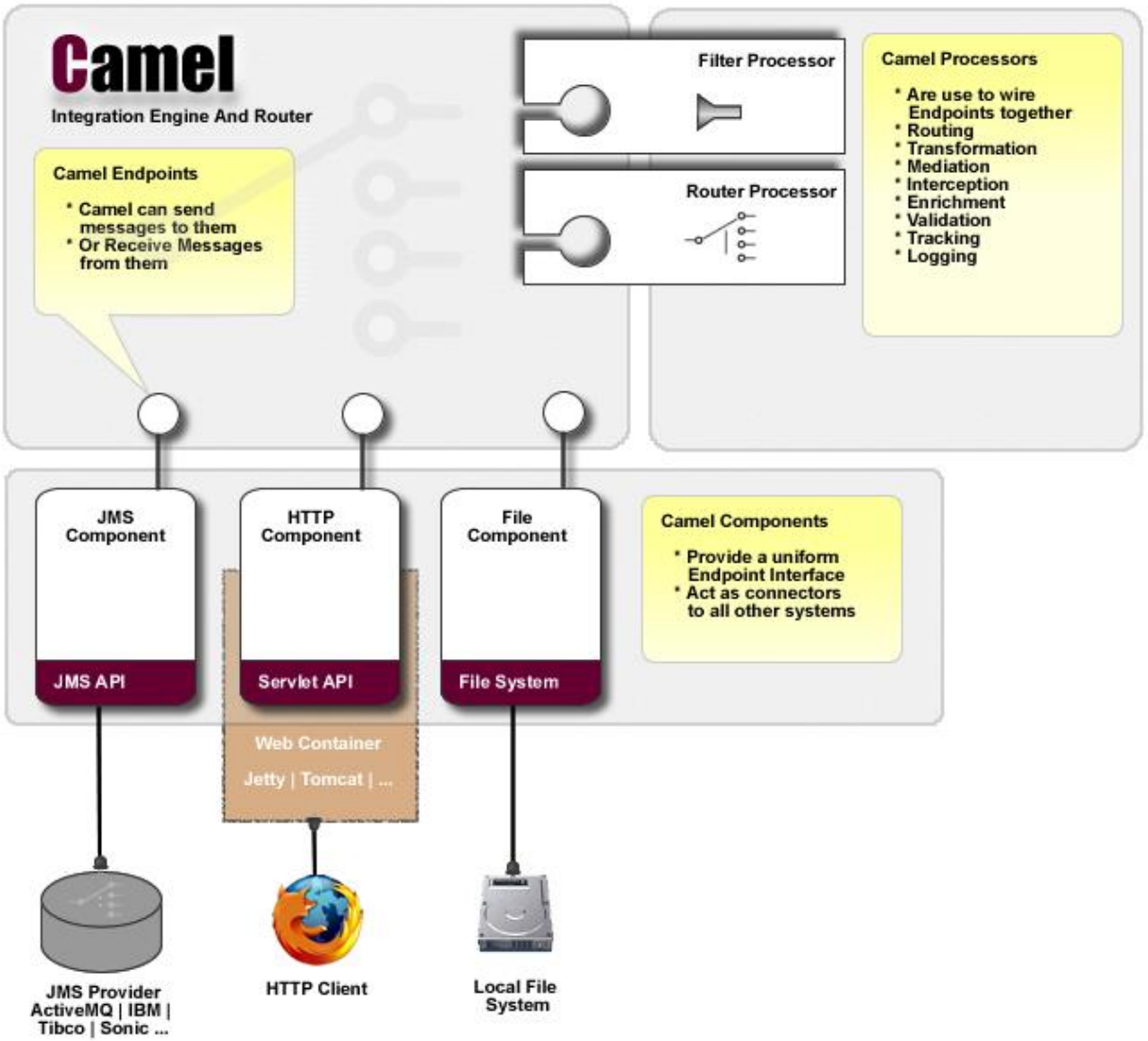
Нода среды выполнения SAP CPI (SAP CPI runtime node)



Apache Camel

- Apache Camel (https://camel.apache.org/) – интеграционный фреймворк (не интеграционная шина!) с открытым исходным кодом, лежащий в основе SAP CPI.
- Поддерживает большинство шаблонов интеграции корпоративных приложений (EIP) (https://www.enterpriseintegrationpatterns.com/).
- Поддерживает предметно-ориентированные языки (DSL) – Java DSL, Blueprint XML, Spring XML, Rest DSL и т.д.
- Поддерживает встраивание в различные сервера приложений и среды выполнения – в т.ч. использование Camel компонентов в OSGi контейнере.
- Поддерживает высокий уровень абстракции и модуляризации.
- Не делает предположений об обрабатываемых данных (payload agnostic).

Apache Camel: архитектура



Источник: https://camel.apache.org/manual/latest/architecture.html

Apache Camel: маршрутизация

- Основа фреймворка – механизм маршрутизации.

- Правила маршрутизации (routing rules) создаются с помощью компоновщика механизмов маршрутизации (route builder) и выполняются в составе наборов правил маршрутизации (routing rulebase) в Camel контексте (Camel context).

- Позволяет описать и определить источники сообщений, правила обработки сообщений и последующей передачи сообщений.

Apache Camel: компоненты и процессоры

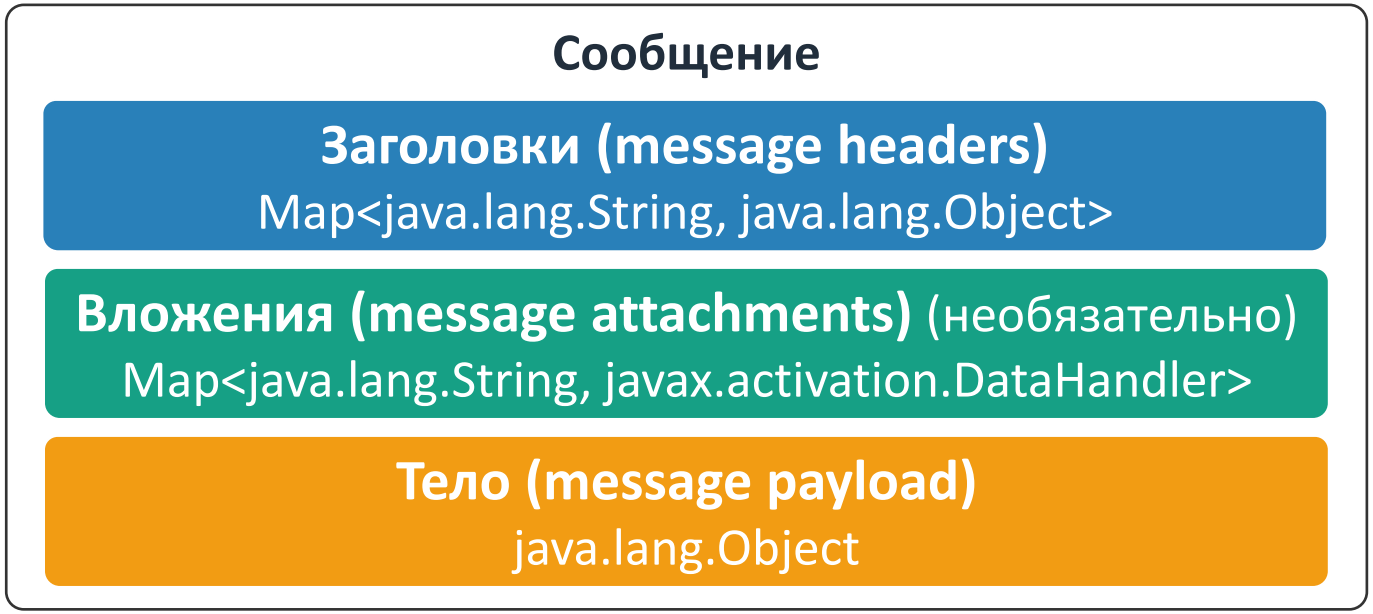
- Ключевые элементы – **компоненты** (components) и **процессоры** (processors).

Компоненты	Процессоры
• Предоставляют функциональность для подключения и взаимодействия с интегрируемыми системами. • Компоненты – это адаптеры или коннекторы, которые используются для создания конечных точек (endpoints).	• Предоставляют функциональность для обработки сообщений. • Примеры: маршрутизация и фильтр сообщений, проверка содержимого сообщений, преобразование форматов данных, шифрование и т.д.

- 280+ компонентов «из коробки», библиотека компонентов может быть расширена сторонними компонентами.

Apache Camel: контейнер и сообщение

- Конечная точка сценария создает экземпляр передачи (exchange) при получении сообщения (message) компонентом-потребителем (consumer). Это контейнер, содержащий информацию на протяжении процесса маршрутизации сообщения. Его метаданные описываются с помощью свойств (exchange properties).
- Структура сообщения:



Интеграционный сценарий (iFlow) в SAP CPI

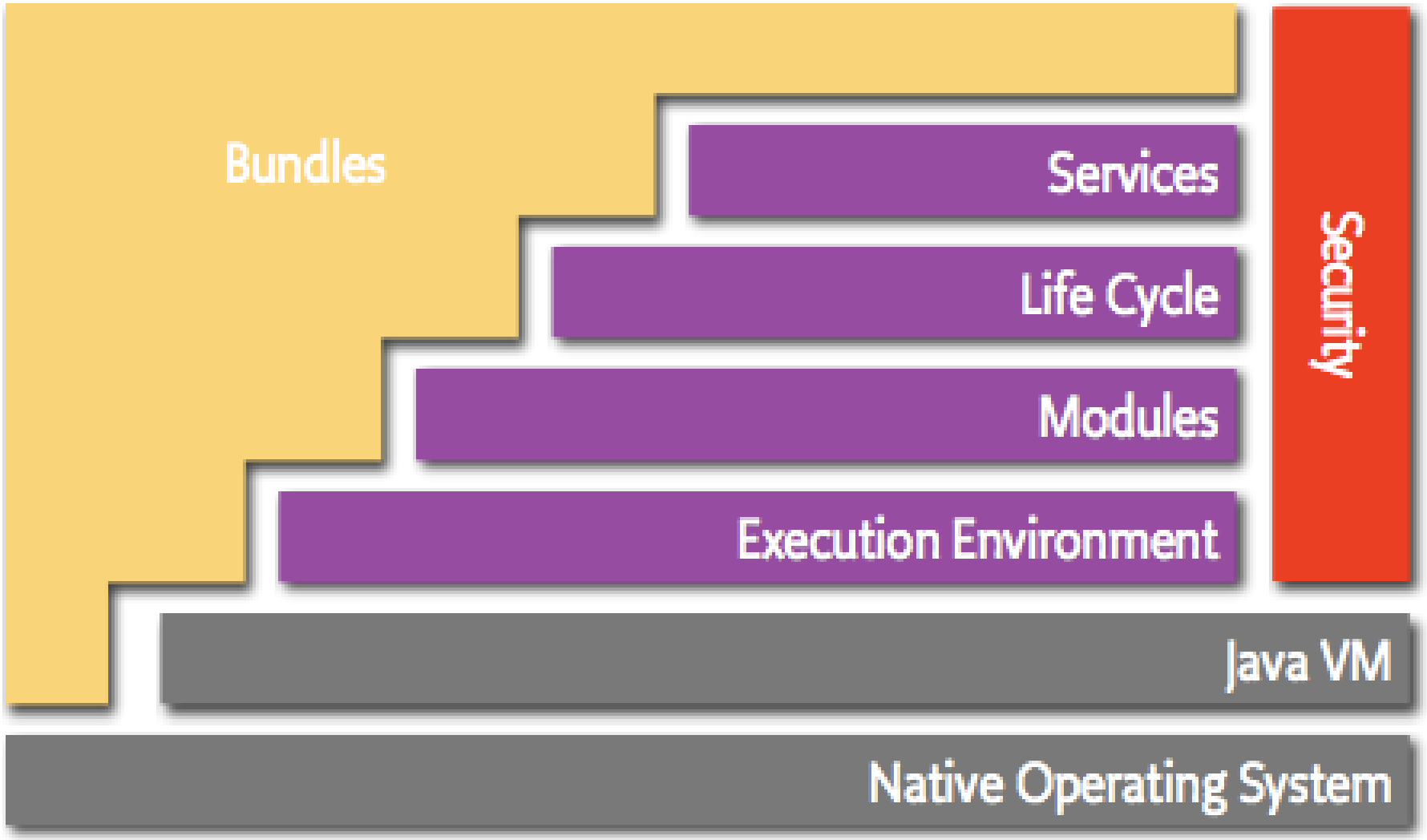
В среде дизайна

- Описывается с использованием Business Process Model and Notation (BPMN).
- Собирается в пакет, содержащий метаданные сценария, его описание, параметры и ресурсы (скрипты, мэппинги, схемы, локальные библиотеки).
- Один интеграционный сценарий (integration flow) может состоять из одного или нескольких интеграционных процессов (integration processes).

В среде выполнения

- Интеграционный процесс в среде дизайна = Camel маршрут в среде выполнения.
- Для каждого интеграционного сценария создается отдельный Camel контекст.
- Если интеграционный сценарий состоит из нескольких интеграционных процессов, соответствующие Camel маршруты добавляются в общий Camel контекст интеграционного сценария.
- Каждый интеграционный сценарий собирается и разворачивается в виде отдельного OSGi Blueprint пакета.

OSGi фреймворк: архитектура



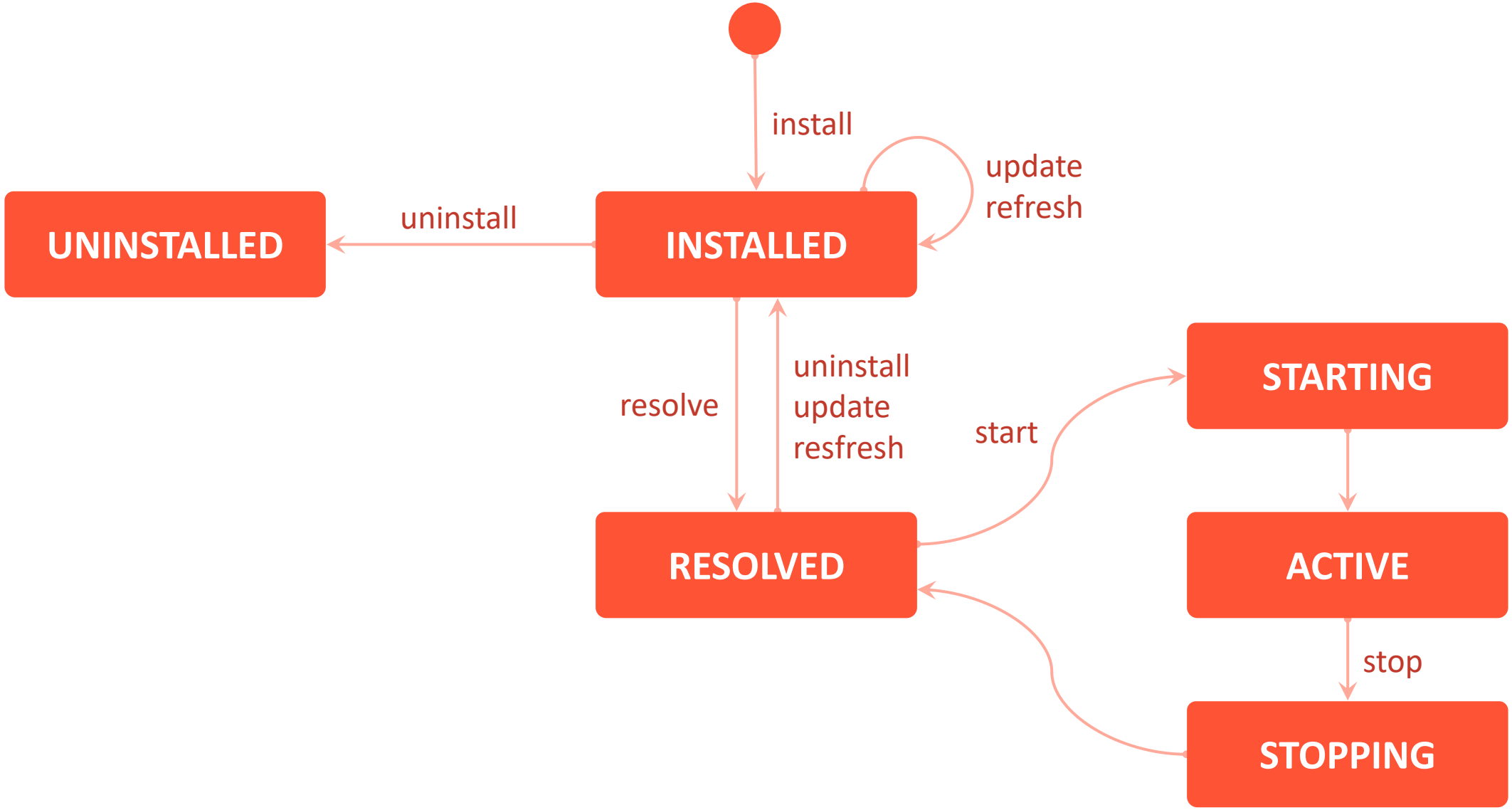
Источник: https://www.osgi.org/developer/architecture/

Модуляризация в OSGi: пакет (bundle)

OSGi пакет

Манифест (META-INF/MANIFEST.MF)	Метаданные пакета, описанные с использованием стандартных и дополнительных заголовков. Например, описание, зависимости (импортируемые ресурсы) и предоставляемые ресурсы (экспортируемые ресурсы).
Ресурсы	Ресурсы, необходимые для предоставления пакетом необходимой функциональности. Например, файлы классов Java, статические ресурсы, локальные для пакета зависимости.
Документация (OSGI-OPT/) (необязательно)	Файлы документации.

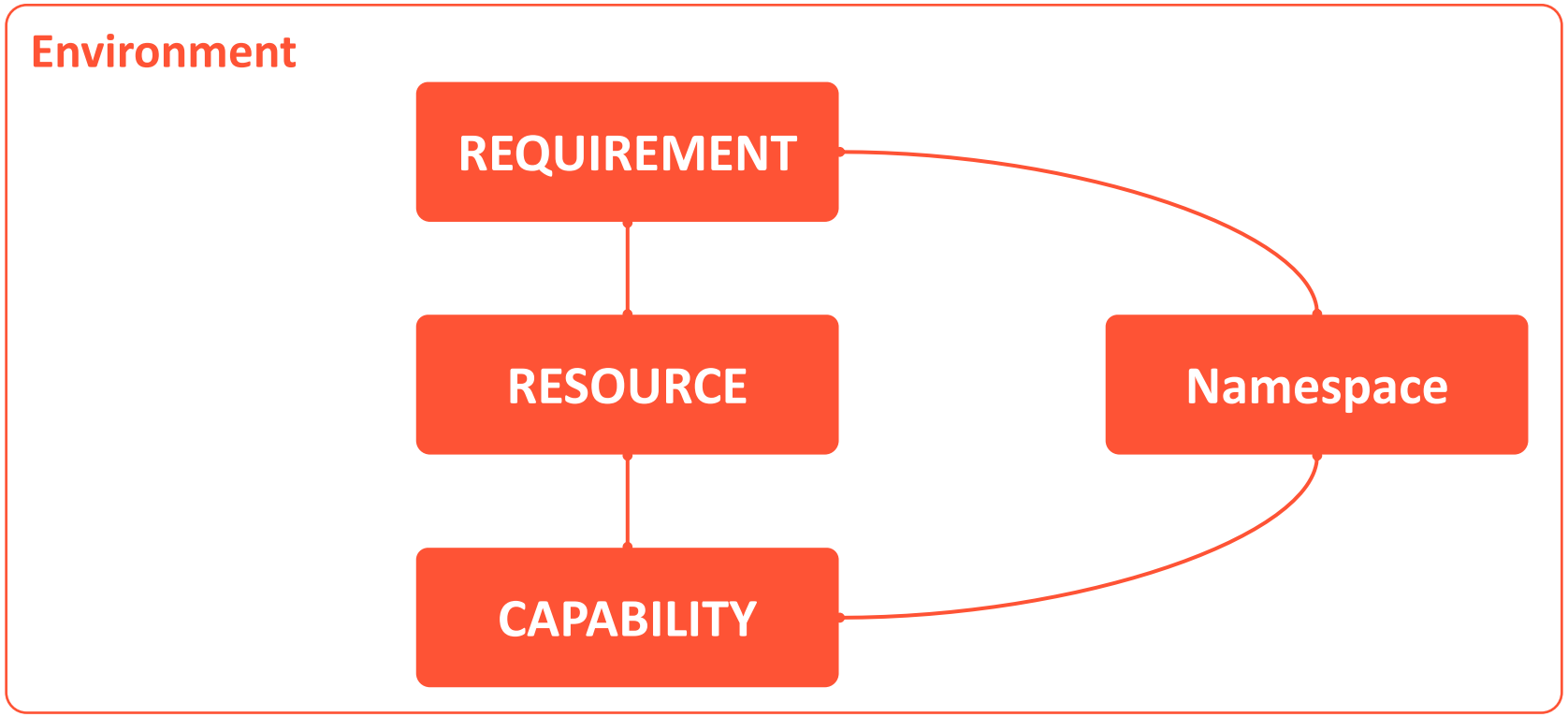
Жизненный цикл пакета



Управление зависимостями: связывание пакетов (bundle resolution & wiring)

- Зависимости описываются в манифесте пакета и определяются как:
  - **Требования пакета** (bundle requirements) – например, указанные в заголовке Import-Package.
  - **Возможности пакета** (bundle capabilities) – например, указанные в заголовке Export-Package.
- Во время разрешения зависимостей пакета (bundle dependencies resolving), требования пакета связываются с найденными и разрешенными возможностями, предоставляемыми другими пакетами, путем создания связи пакетов (bundle wire).

Управление зависимостями: связывание пакетов (bundle resolution & wiring)



Загрузка классов (class loading)

- Стратегия поиска классов – поиск в глубину (depth-first search).
- Пространство классов (классы, доступные из загрузчика классов пакета):
  - Родительский загрузчик классов (parent class loader / boot class path) – для классов, содержащихся в пакетах java.*.
  - Импортированные пакеты (imported packages).
  - Требуемые пакеты (required bundles).
  - Путь к классам пакета (bundle’s class path / private packages).
  - Прикрепленные фрагменты (attached fragments).

Спасибо за внимание

Вадим Климов

Архитектор интеграционных решений SAP

https://people.sap.com/Vadim.Klimov