LUXOFT TRAINING

Использование платформы Cloudify PaaS для ускорения разработки приложений

Mikhail Druzhinin, SECR, 2013

Информация о спикере

- Дружинин Михаил
- Architect, Head of Cloud CoE

- Контакты:
  - E-mail: mdruzhinin@luxoft.com

О чём пойдёт речь?

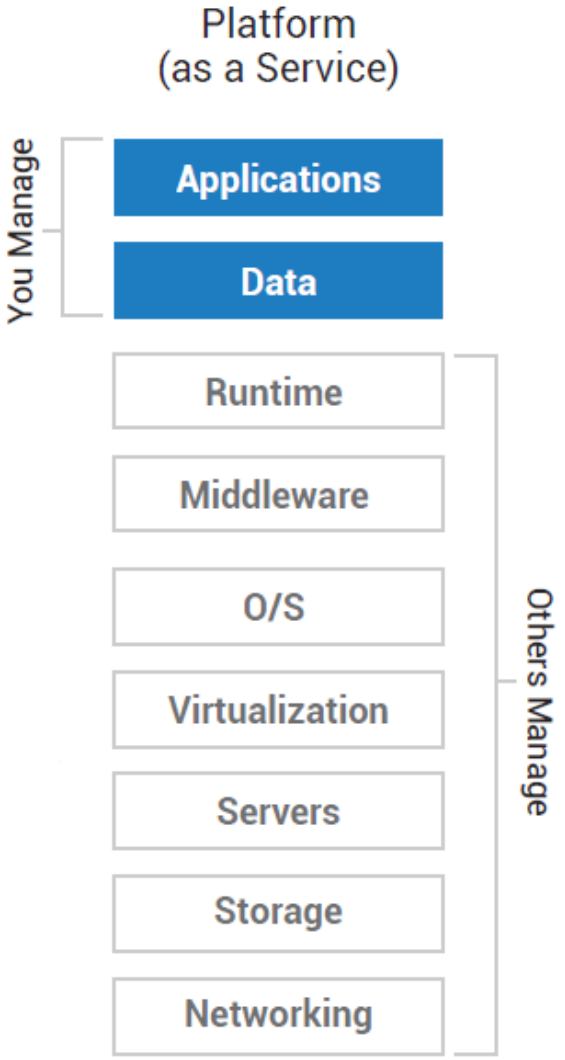
- Жизнь разработчика
- Что такое PaaS?
- Какие проблемы решает PaaS?
- Архитектура Cloudify PaaS
- Как начать?
- Архитектурные особенности приложения под PaaS
- Плюсы и минусы

Жизнь разработчика

- Сделал приложение (war)
- Собрал приложение
- Запустил у себя – всё отлично
- Запустил в тестовом окружении – отдал в тестирование
- …

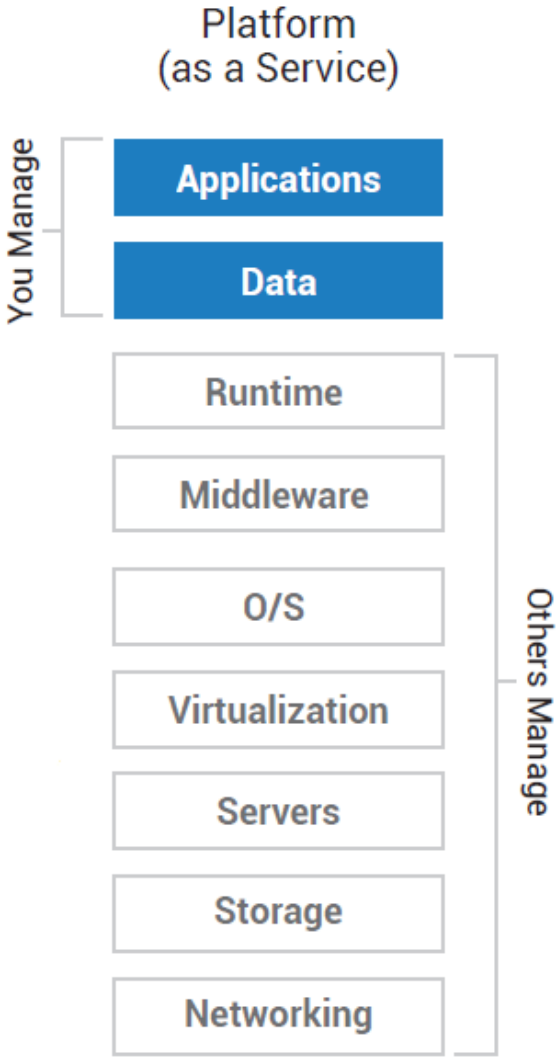
Что такое PaaS?

- Platform-as-a-Service
- Предоставление вычислительных средств и «программного стека» как сервиса
  - Сервера приложений / среды выполнения (middleware)
  - Специализированные вычисления
  - База данных и системы хранения



Чем PaaS не является?

- PaaS это не «облако» в традиционном его понимании
  - Оно предоставляет уже готовые среды выполнения
- PaaS это не application
  - Оно не несёт в себе логики приложения



Плюсы PaaS

- **Скорость:** разработчики получают необходимые ресурсы по запросу
- **Стоимость:** не тратим деньги на менеджмент software и hardware
- **Масштабируемость**

Чего стоит ожидать от PaaS

- Поддержка различных языков, фреймворков и сред выполнения
- Поддержка различных инфраструктур для развёртывания
- Расширяемость
- Автомасштабируемость
- No vendor lock-in

Какие проблемы решает PaaS?

- Автоматизированное развёртывание
- Мониторинг параметров системы
- Автоматическое масштабирование
- Обеспечение восстановления после сбоев
- Одинаковое развёртывание тестовых и боевых окружений

Какие PaaS бывают?

- Public / BlackBox



- Private

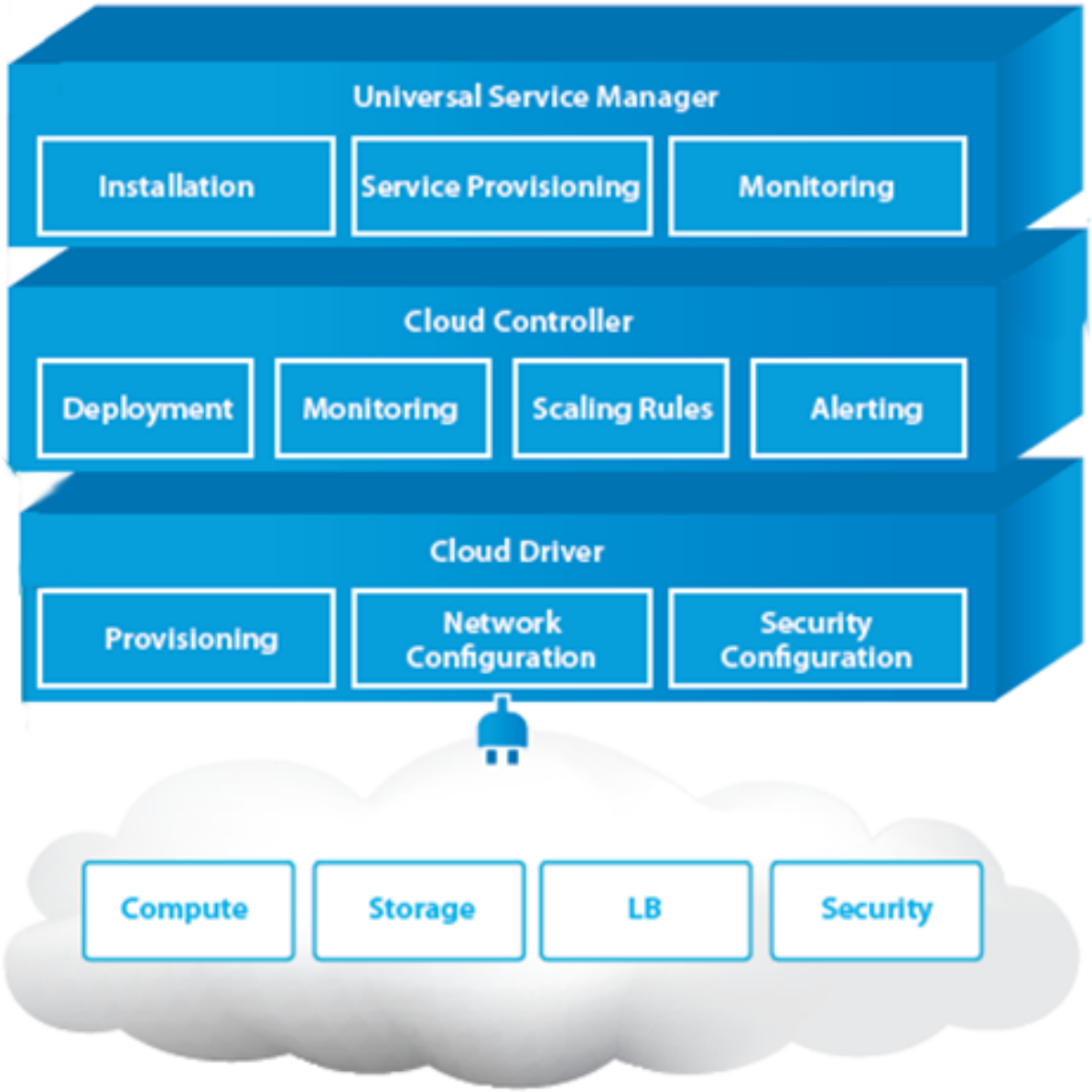
Cloudify PaaS

- Архитектура Cloudify
- Как оно работает
- С чего начать?
- Рецепты развёртывания
- Интеграция с chef

Архитектура Cloudify PaaS

- Рецепты развёртывания
- Управляющий сервер
- «Драйверы» облачных приложений

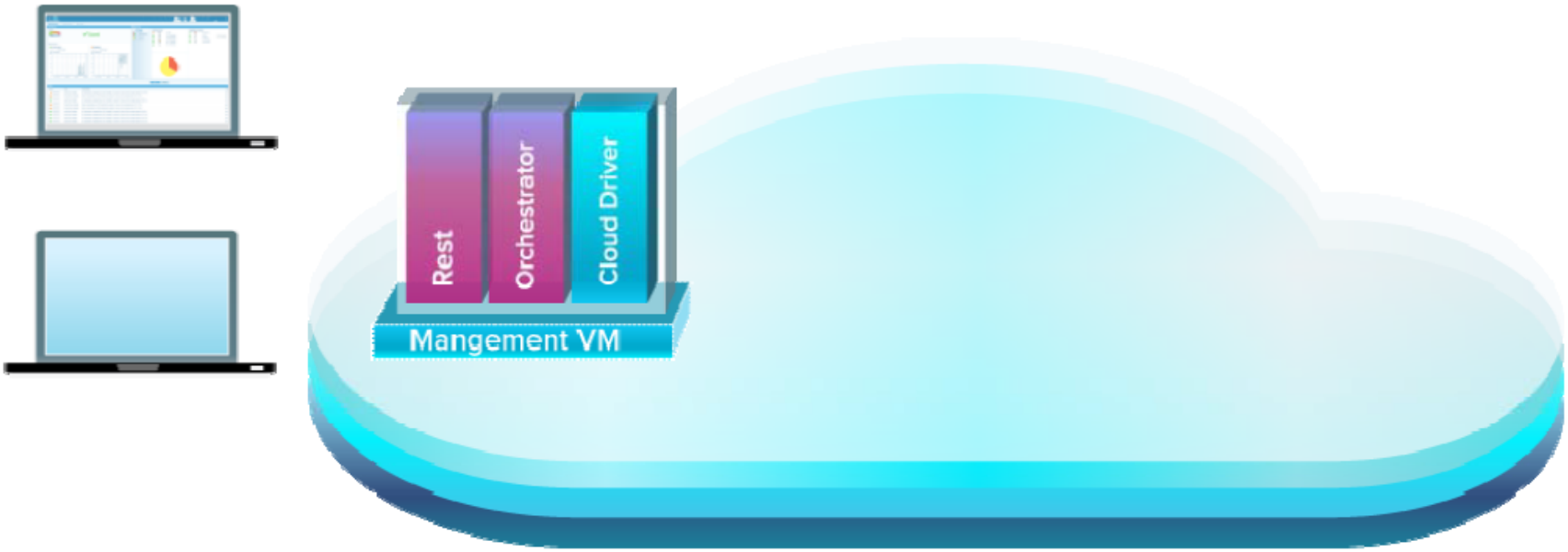
Архитектура Cloudify PaaS



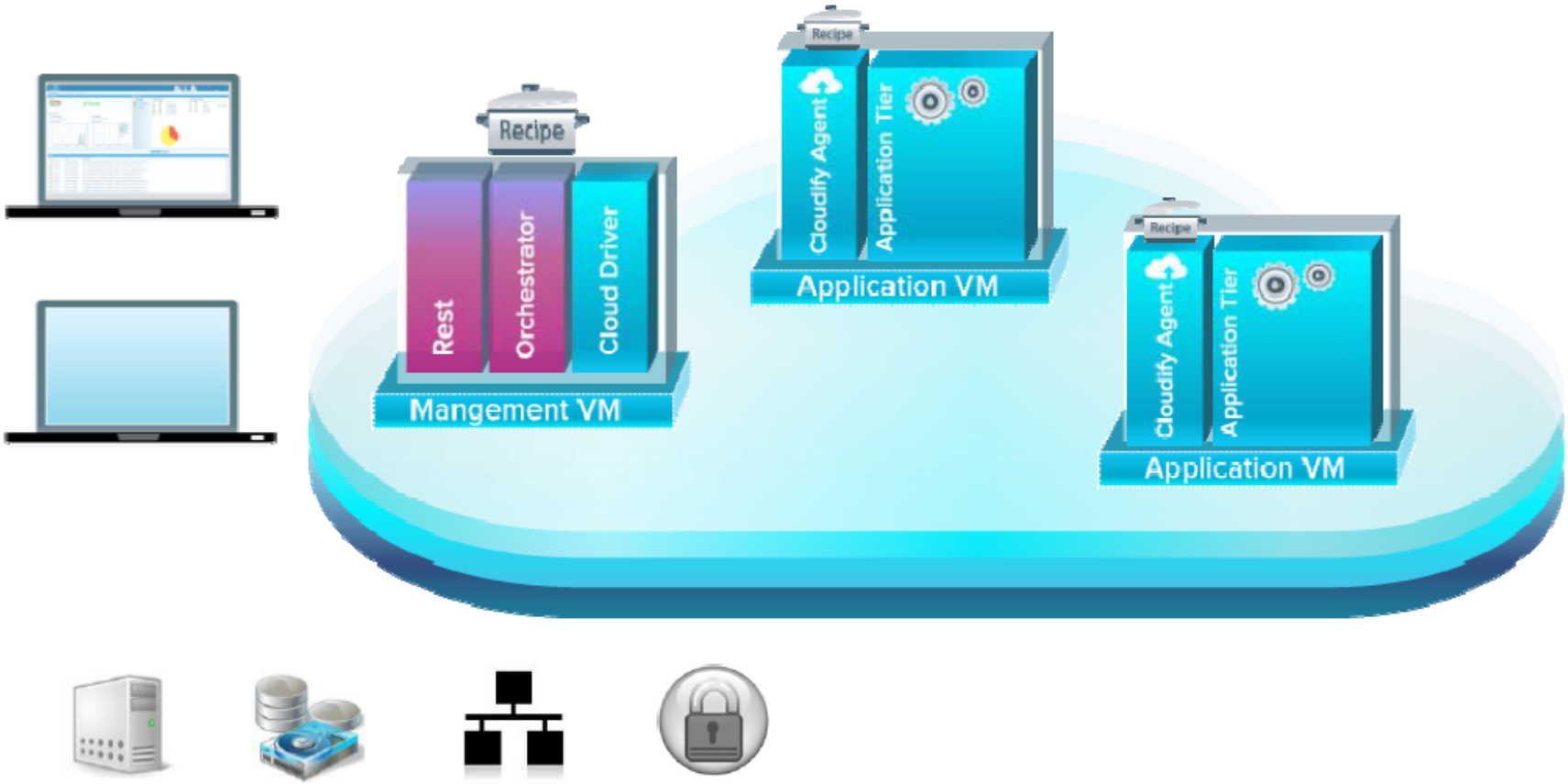
Как оно работает

- Загрузка рецептов
- Запуск управляющей машины
- Создание виртуальных машин
- Установка приложения
- Мониторинг и масштабирование

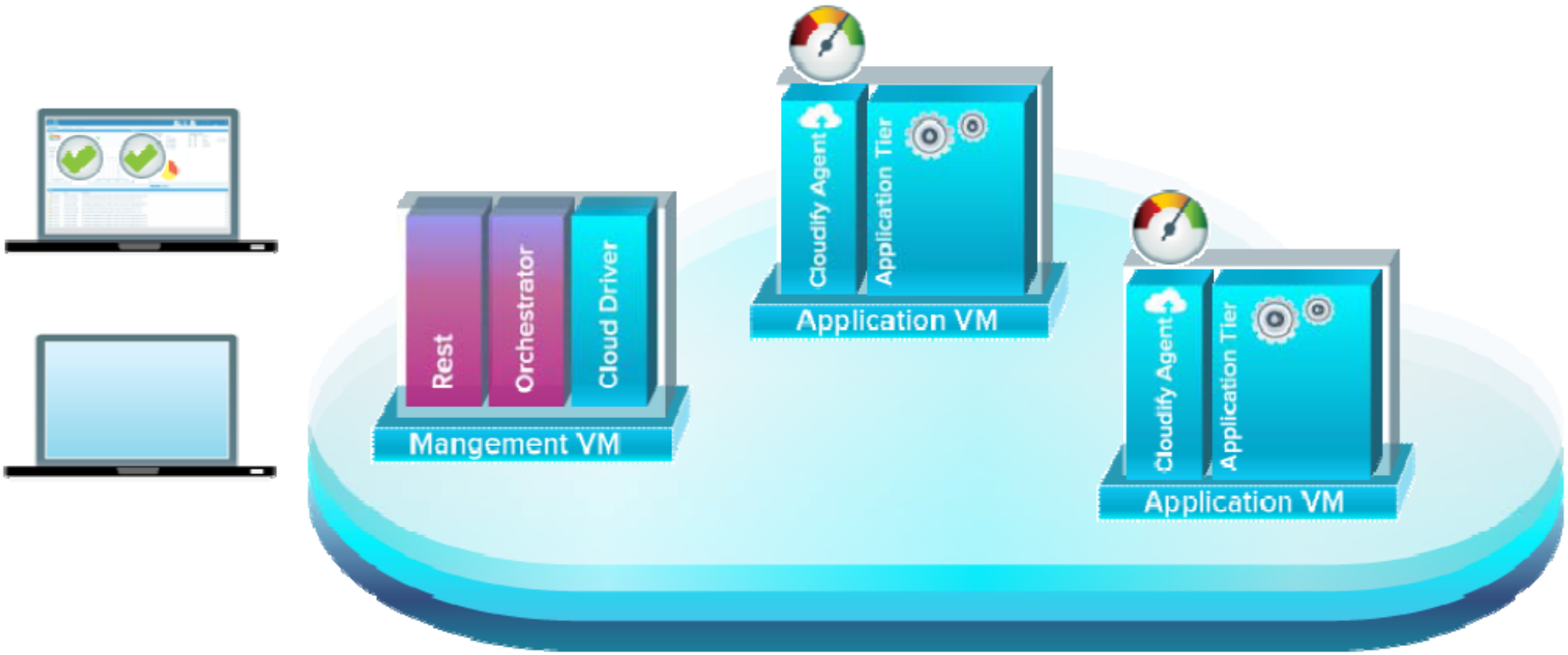
Как оно работает - запуск



Как оно работает - запуск



Как оно работает - мониторинг



Как начать?

- Запуск «облака»
  - `>bootstrap-localcloud`
- Инсталяция приложения
  - `>install-application petclinic`

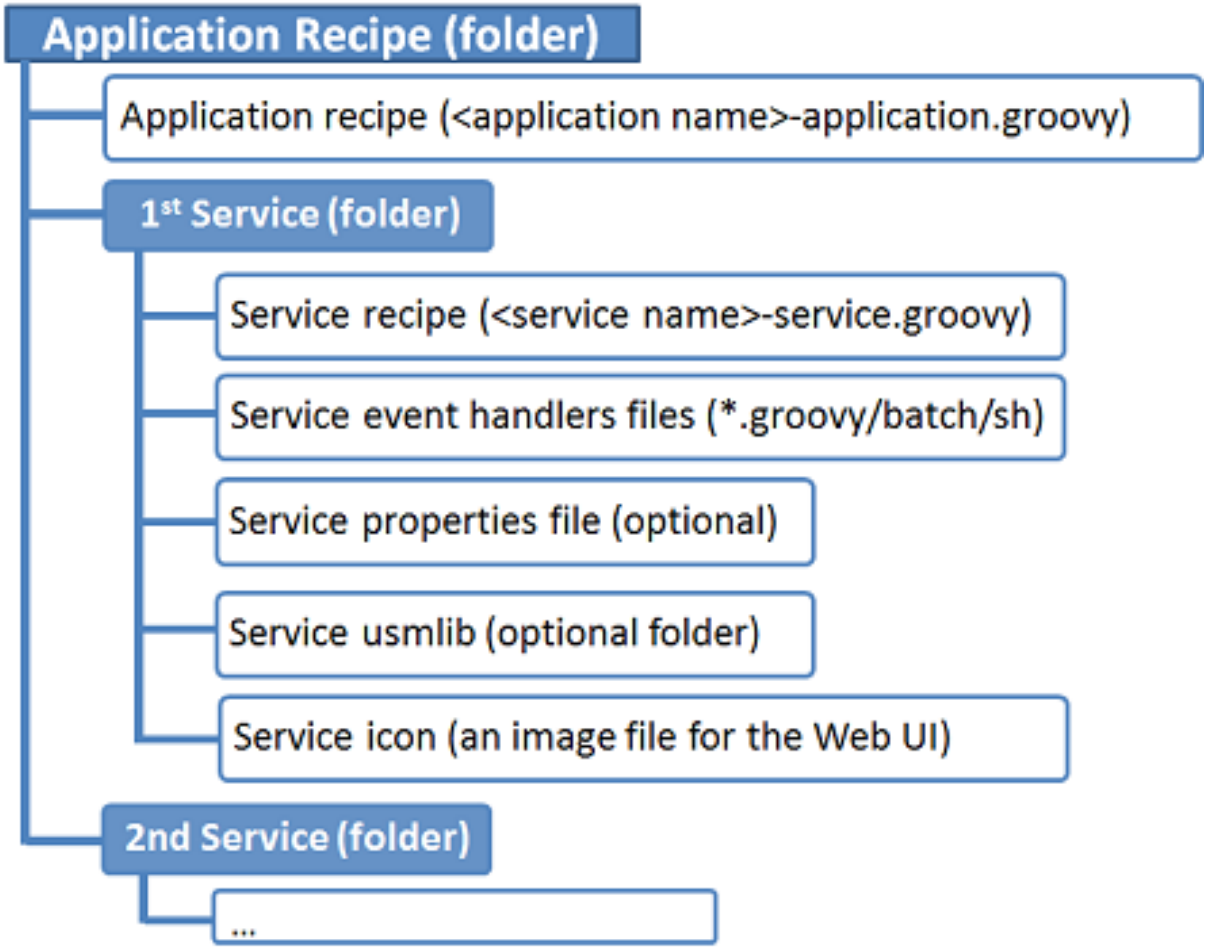
Как начать на Amazon WS?

- Запуск «облака»
  - `>bootstrap-cloud ec2`
- Инсталяция приложения
  - `>install-application petclinic`

Рецепты развёртывания

- Структура рецепта приложения
- Структура рецепта сервиса

Рецепты развёртывания



Рецепты развёртывания - приложение

```
application {
    name="petclinic"
    service {
      name = "mysql"
    }
    service {
      name = "tomcat"
      dependsOn = ["mysql"]
    }
}
```

Рецепты развёртывания - сервис

```
service {
    name "mysql"
    icon "mysql.jpg"
    type "DATABASE"

    lifecycle{
      install "mysql_install.groovy"
      start "mysql_start.groovy"
      ...
    }
}
```

Рецепты развёртывания - масштабирование

```
scalingRule {
    serviceStatistics {
      metric "Requests per second"
      movingTimeRangeInSeconds 20
    }

    highThreshold {
      value 100
      instancesIncrease 1
    }
}
```

Рецепты развёртывания - мониторинг

```
monitors {
  def mBean = "Catalina:type=ThreadPool,name=
                  http-bio-${currHttpPort}"

  def metrics = [
       "Current Http Threads Busy":
       [$mBean, "currentThreadsBusy"],
      "Current Http Threads Count":
       [$mBean, "currentThreadsCount"],
  ]

  return
     getJmxMetrics("127.0.0.1",currJmxPort,metrics)
}
```

Рецепты развёртывания – динамическая конфигурация

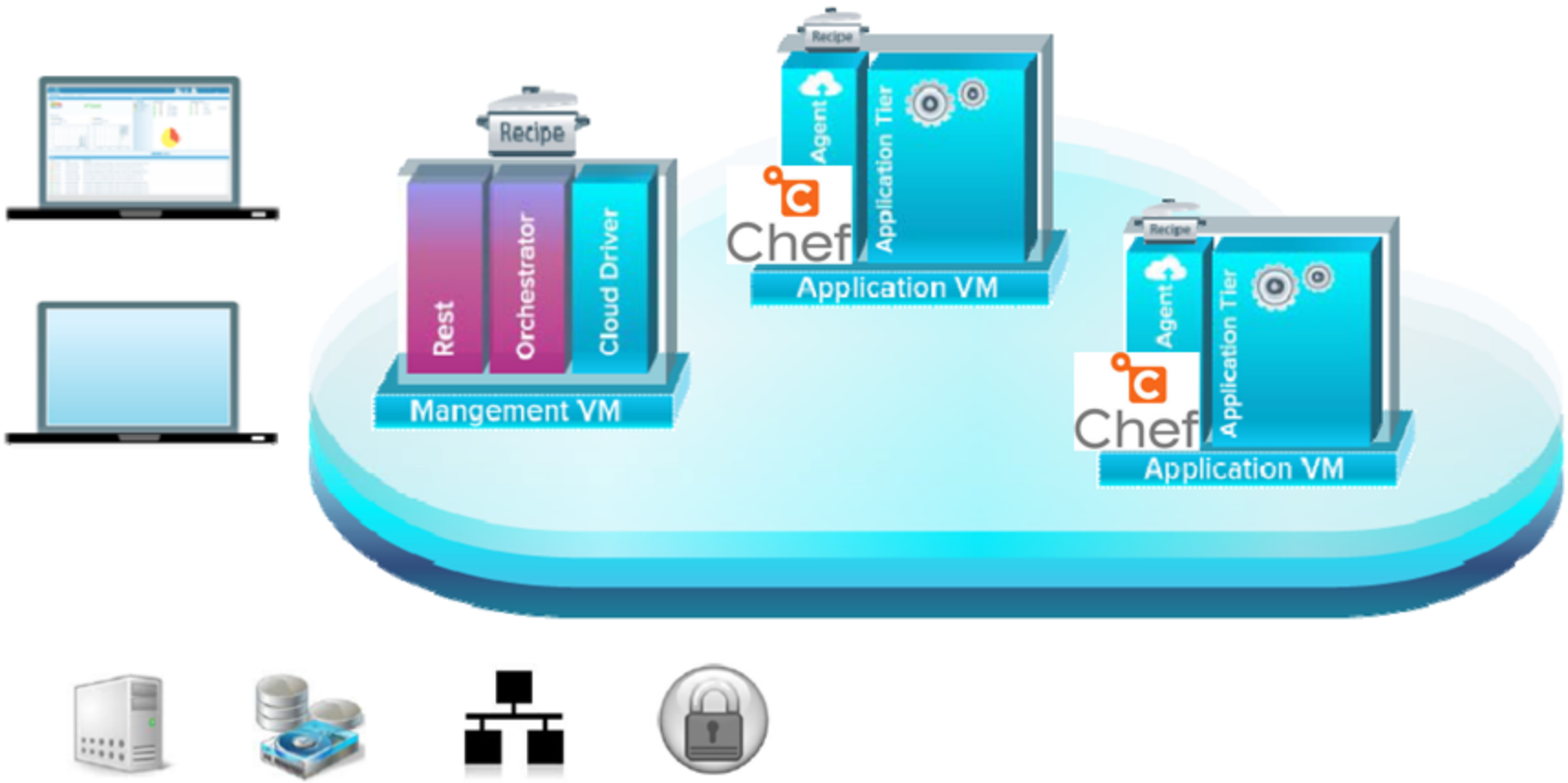
```
serviceContext.attributes
    .thisInstance["port"] = config.port

dbServiceInstances = serviceContext
    .attributes[“db”].instances
```

Рецепты развёртывания – динамическая конфигурация

```
customCommands ([
 "addNode" : "apacheLB_addNode.groovy",
 "removeNode" : "apacheLB_removeNode",
 "load" : "apacheLB-load.groovy"
])
```

Интеграция с Chef



Архитектурные особенности приложения

- Автоматическое обнаружение сервисов
- Любой узел может упасть

Плюсы

- Очень быстрое развёртывание
- Мониторинг и автомасштабирование из коробки
- Не требует изменения приложения
- Легкая доработка рецептов и лёгкое создание сложных сценариев развёртывания
- Возможность работы с различными IaaS / BYON

И минусы

- Мало «родных» рецептов (решается интеграцией с chef)
- Требует отдельную VM для управления
- Придётся самому позаботится о месте хранения артефактов для развёртывания

LUXOFT TRAINING

Благодарю за внимание!

Вопросы
?