SOUTH-EAST EUROPEAN RESEARCH CENTRE

SEERC

The University Of Sheffield.

CITY College An International Faculty Of The University.

A Development Framework Enabling the Design of Service-based Cloud Applications

Fotis Gonidis

South East European Research Centre,
Thessaloniki, Greece

Outline

- Introduction
  - Cloud Application Platforms
  - Service-based Cloud Applications
- Development Framework
  - High-Level Architecture
- The case of Payment Service
- Conclusions

Introduction

Software as a Service

Platform as a Service

Infrastructure as a Service

- Focus on designing cloud applications, deployed in cloud platforms

Introduction

Platform classification into 3 categories

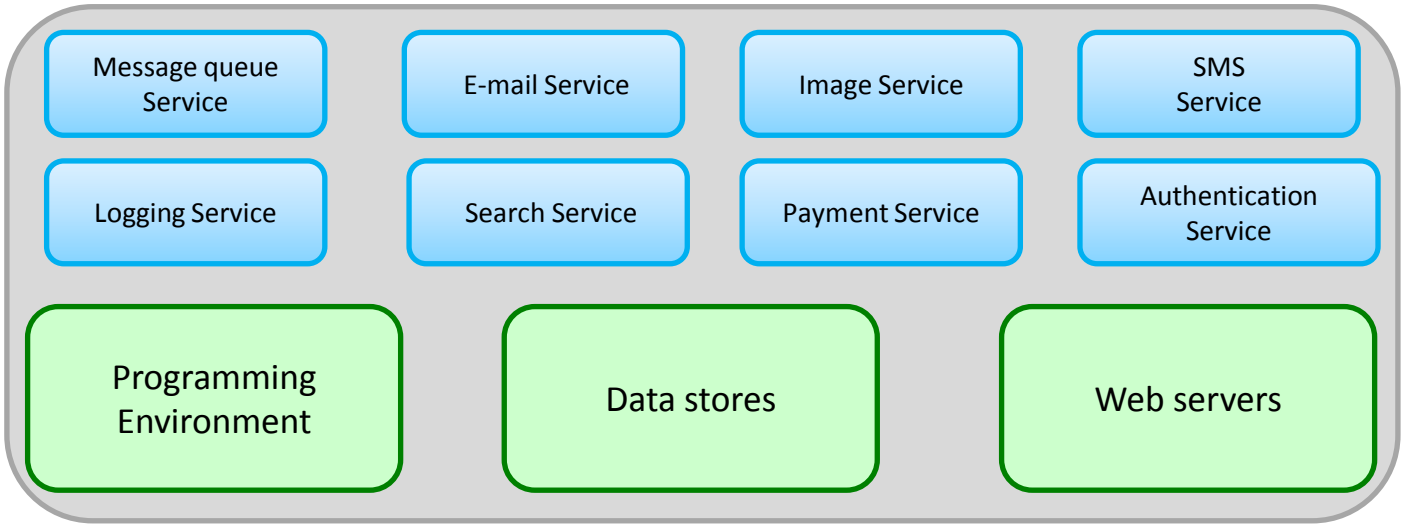
- Support for widely used technologies: SQL, Java, PhP, etc
- Generic application scope, high application development time
- Low level lock-in effect

- Support for widely used technologies: SQL, Java, PhP, etc
- Generic application scope, medium application development time
- Medium level lock-in effect
- Additional functionality through platform services via API

- No support for widely used technologies
- Specific application scope
- Short application development time
- High level lock-in effect
- Online development via web-browser and visual interfaces

Introduction

Cloud Application Platform[1]

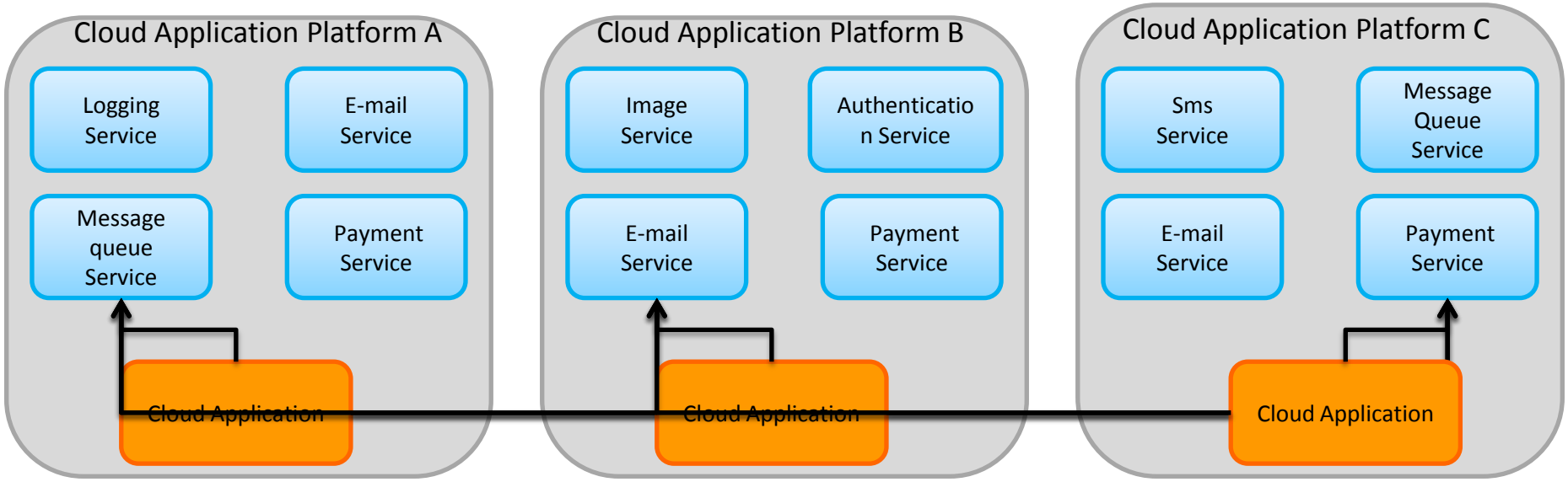


- Kourtesis et al. puts forwards the idea of cloud application platforms. Provisioning of traditional platform resources **and platform services**: e-mail, message queue, sms, authentication, payment service etc.
- Cloud applications are composed of multiple platform services. Therefore
  - Reduced application development time, upfront and maintenance costs, increased functionality

1. Dimitrios Kourtesis, Konstantinos Bratanis, Dimitris Bibikas, Iraklis Paraskakis: **Software Co-development in the Era of Cloud Application Platforms and Ecosystems: The case of CAST**. PRO-VE 2012: 196-204

Introduction

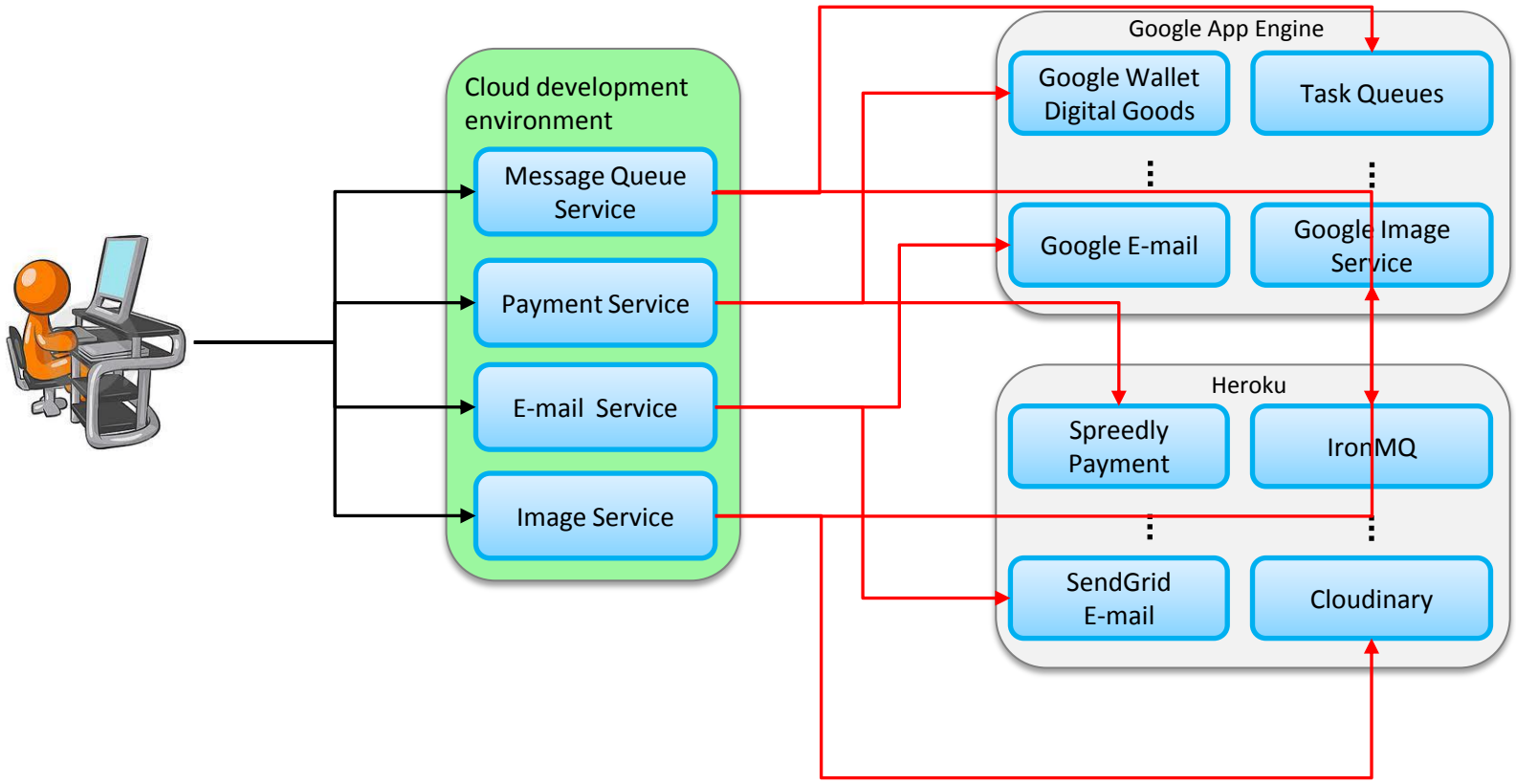
Service-based Cloud Application



- Service-based cloud application is deployed in a cloud application platform and use a combination of platform services
- Plethora of platform services and service providers, need for seamless use of the services

Introduction

Cloud Application Development



- Design of service-based cloud application independent from the concrete service providers

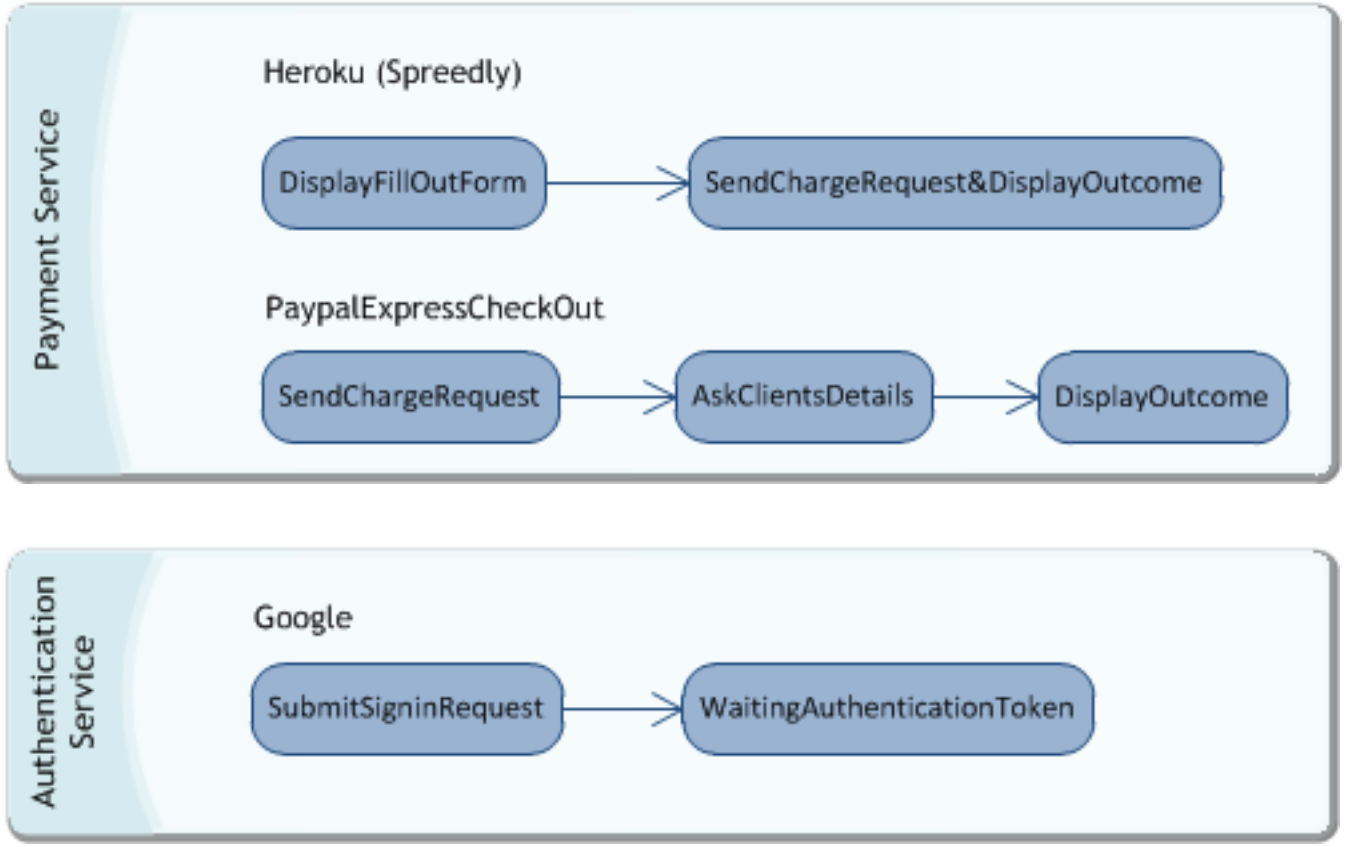
Consuming Platform Services

Variability points in consuming Platform Services

- Different workflows required to complete the operations
- Differences in the web API exposed by the service providers
- Managing the configuration settings, the credentials and the authentication tokens required by each service provider

Consuming Platform Services

Handling the workflow required to complete an operation



Consuming Platform Services

Deal with the service provider`s web API

- E-mail Service
  send an e-mail

Heroku (SendGrid)	from	to	subject	text
Amazon Simple E-mail Service	Source	Destination. ToAddresses	Message. Subject	Message.Body. Text
Heroku (PostMark)	From	To	Subject	TextBody

- Payment Service
  send a payment transaction request

Heroku (Spreedly)	amount	Payment_method_ token	currency_code
Amazon (Stripe)	amount	card	currency

Consuming Platform Services

Managing configuration settings, credentials and authentication tokens

- Payment service

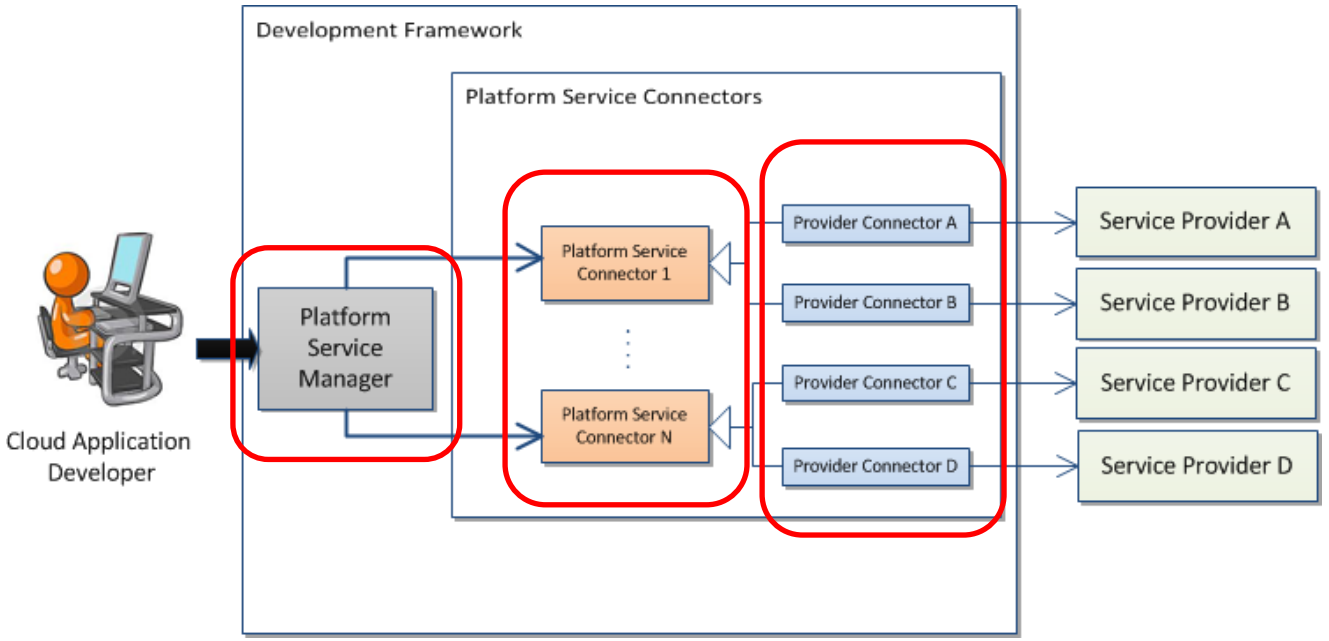
Heroku (Spreedly)	Username	Password	Redirect URL	Gateway Token

- Authentication service

Google Authentication	Client_Id	Client_Secret	State	Scope

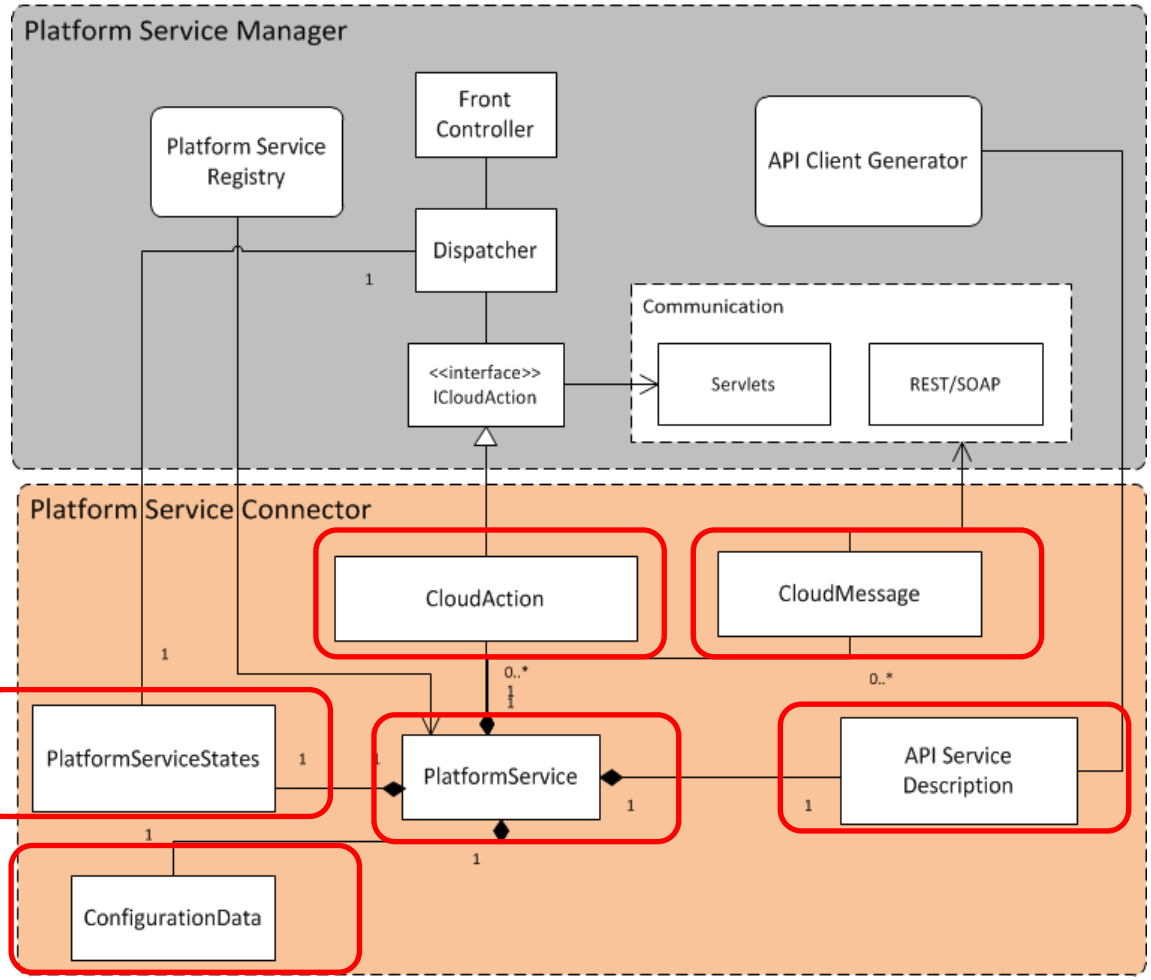
Development Framework

High-level Architecture



- **Platform Service Manager**: coordinates the interaction between the application and the platform services
- **Platform Service Connector**: contains a reference abstract description of the functionality of the service. One connector for each platform service
- **Provider Connector**: detailed implementation for each concrete service provider

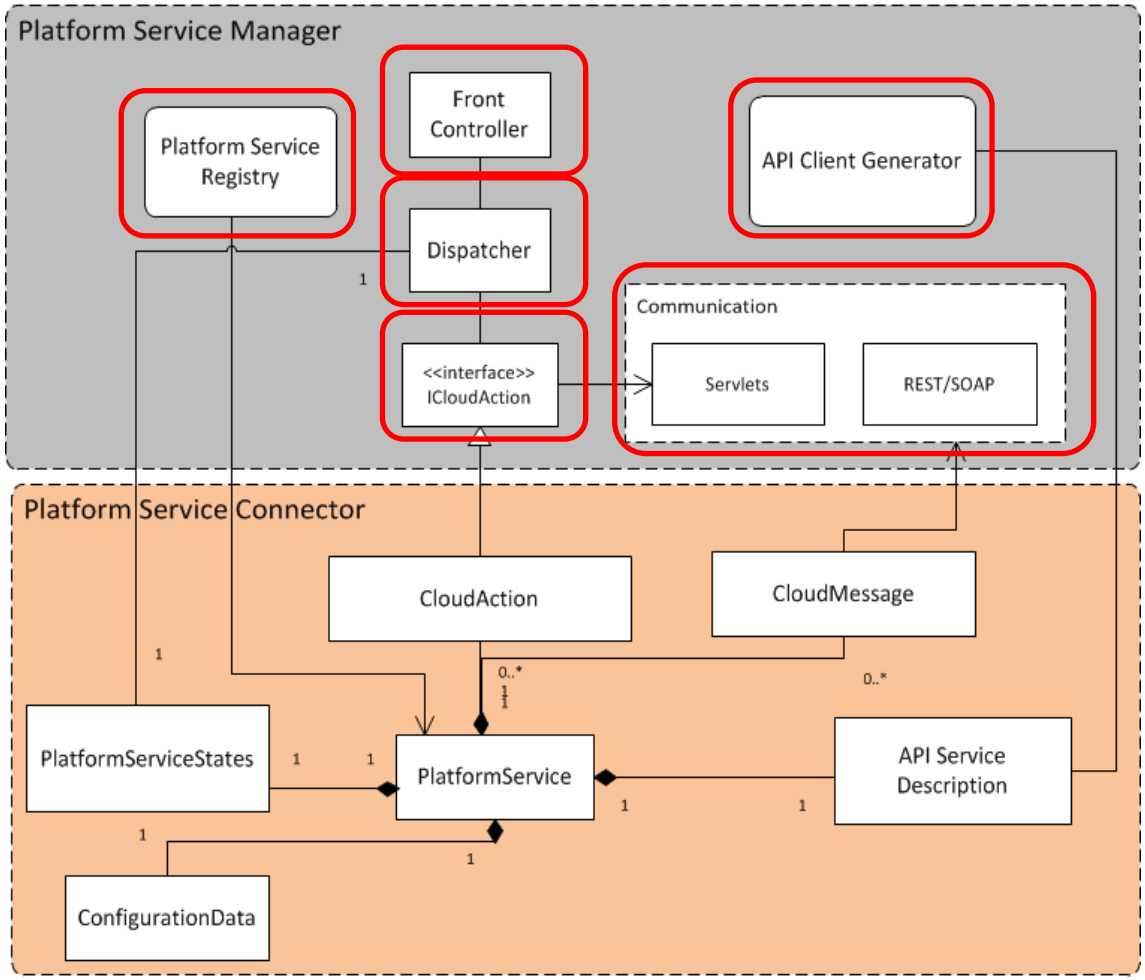
Development Framework



Meta-model

- **Cloud Action**
  Handles the incoming requests
- **Cloud Message**
  Sends requests via web API
- **Platform Service States**
  Lists the states and the action to be invoked
- **ConfigurationData**
  Settings, credentials, tokens
- **API Service Description**
  Describes the web API
- **Platform Service**
  Packages the components of the Platform Service Connector

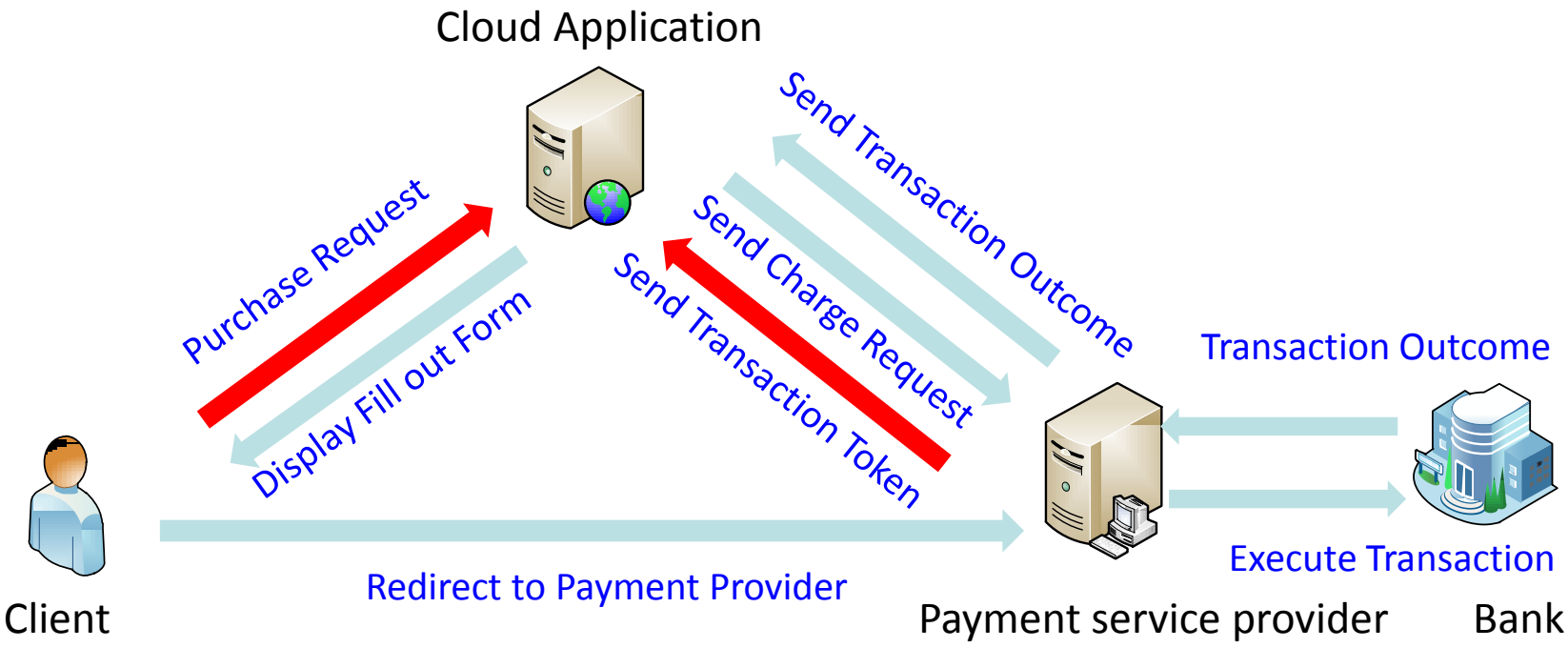
Development Framework

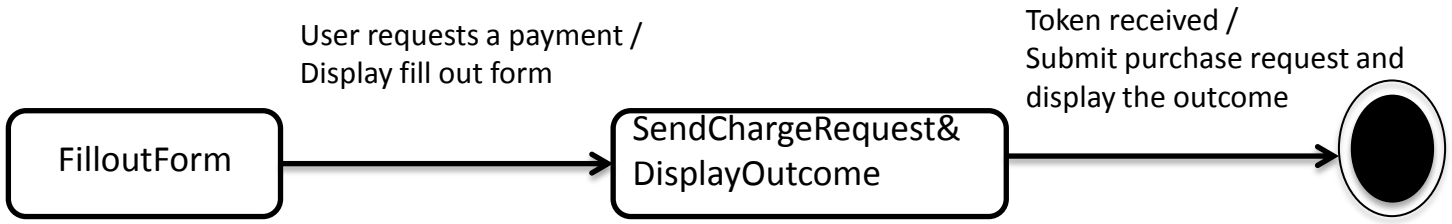


Platform Service Manager

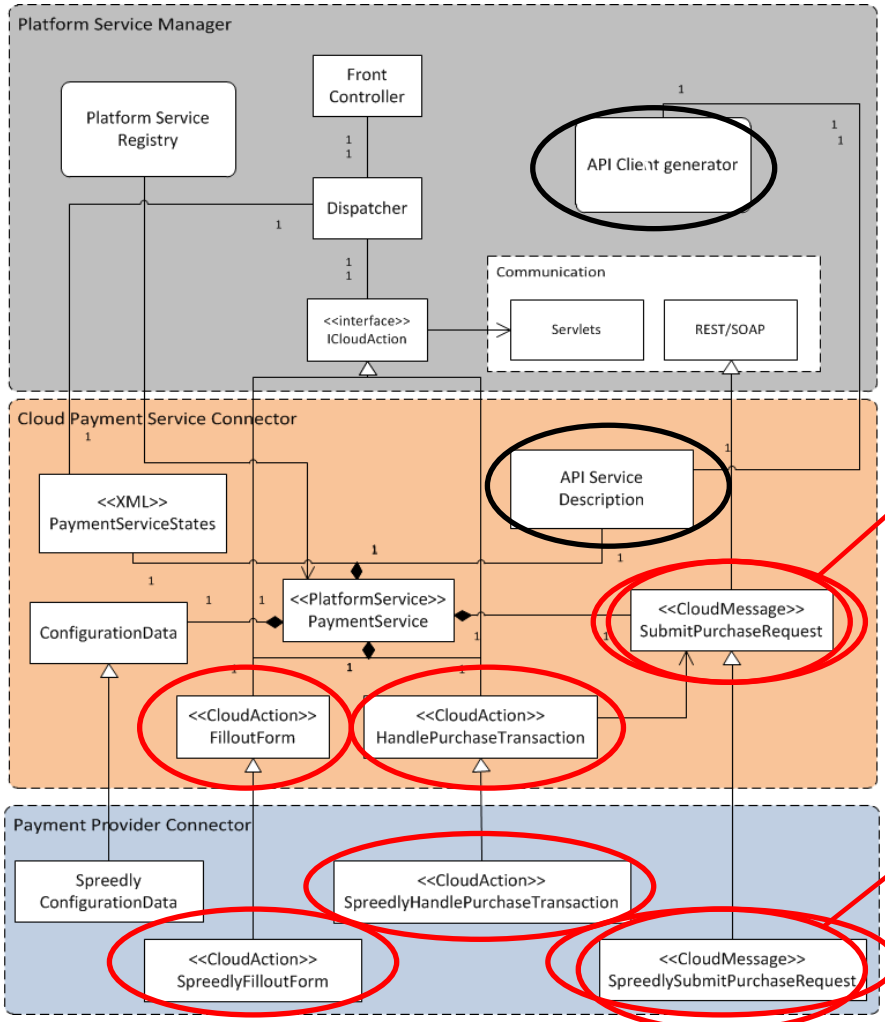
- **Front Controller**
  Receives incoming requests
- **Dispatcher**
  Calls the CloudAction to handle the request
- **ICloudAction**
  Interface to the CloudAction
- **Communication**
  Servlet, REST/SOAP implementation
- **Platform Service Registry**
  Keeps track of the registered services
- **API Client Generator**
  Generates provider specific adapters

The case of Payment Service





The case of Payment Service



- Map the states on the Meta-model
  - For each state, a CloudAction is defined to handle the request
  - For each web request from the application to the service, a CloudMessage is defined
  - C
  - E

Reference model

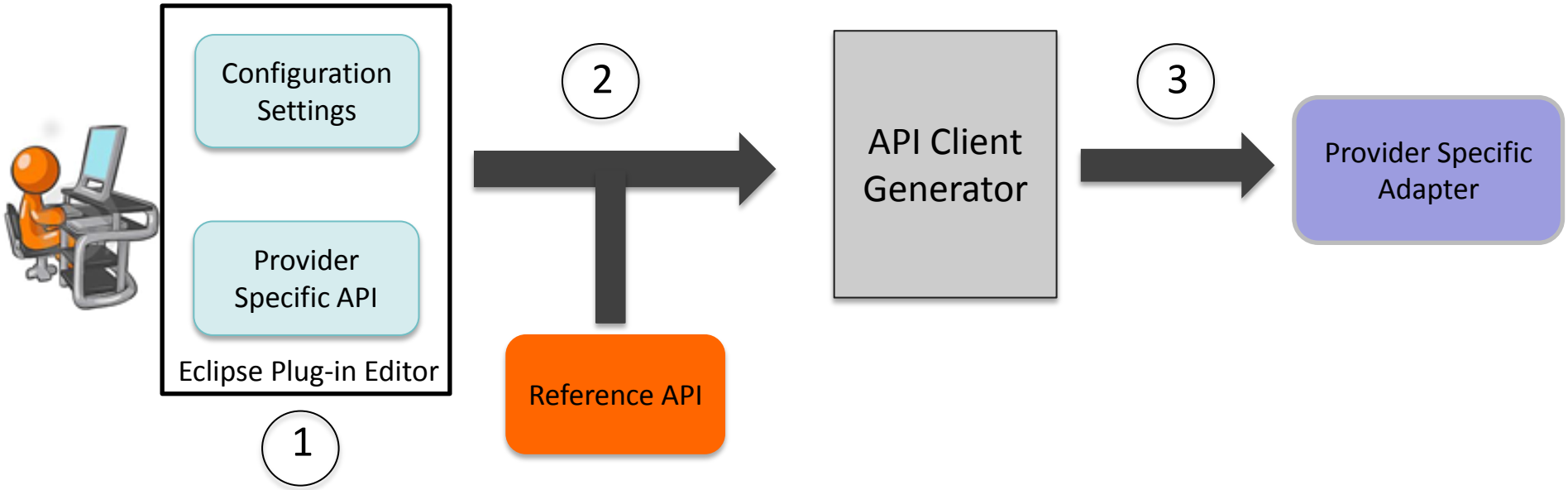
Purchase Request	Currency Code	Card Identifier	Charged Amount

- Add Provider specific implementation
  - Add the specific CloudActions and CloudMessages
  - Edit the PlatformServiceStates file and the ConfigurationData

Purchase Request	currency_ code	card	amount

Heroku (Spreedly)

API Service Description



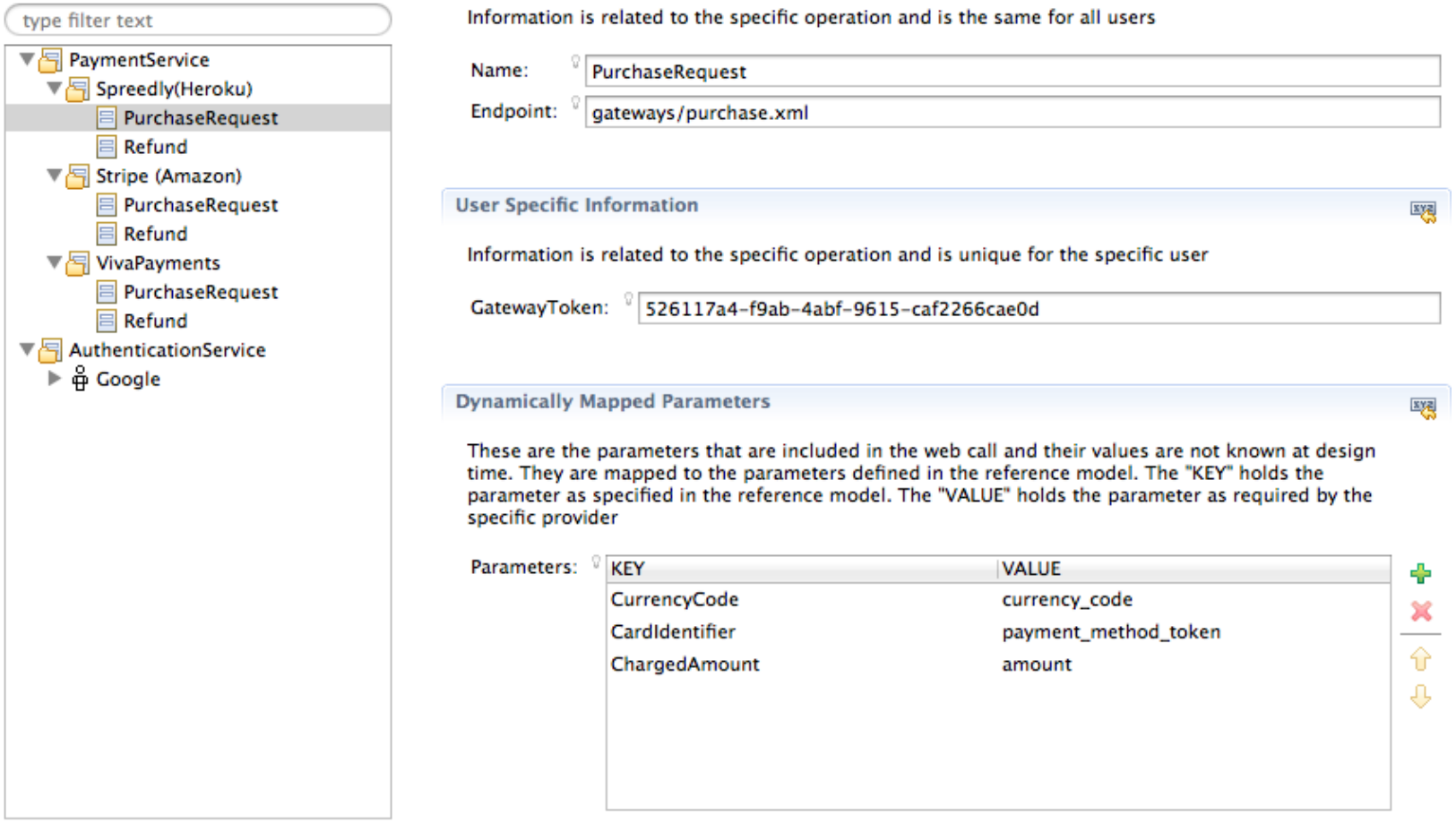
1. The user of the framework describes the Provider Specific API and the Configuration Settings
2. The Provider Specific API is mapped on the Reference API used by the application
3. The API Client Generator, based on the service provider description, generates the Provider Specific Adapter

SOUTH-EAST EUROPEAN RESEARCH CENTRE

The University Of Sheffield.

CITY College An International Faculty Of The University.

API Service Description



Conclusions

- Plethora of cloud application platforms
- Service-based cloud application are becoming increasingly popular
  Need for seamless and transparent use of platform services

- 3 points to consider when consuming platform services
  - Handle the flow to complete an operation
  - Deal with the web API exposed by the service providers
  - Manage the configuration settings, credentials, tokens for each service provider

Future Work

- Refine the API service description. Investigate the role of ontologies for the semantic description of the services.

SOUTH-EAST EUROPEAN RESEARCH CENTRE

The University Of Sheffield.

CITY College An International Faculty Of The University.

Thank you

E-mail: fgonidis@seerc.org
LinkedIn: Fotis Gonidis
Tel: +30 2310 253477