**GetO2PUK**

**Design Document V1.2**

|  |  |
| --- | --- |
| Product | GetO2PUK Application Design Document |
| Code |  |
| Zone |  |
| Description | Web Based Application that enables O2 customer to obtain their PUK code |
| Status | V1.2 |

Table of Contents

[1. Document Control 3](#_Toc469918683)

[1.1 Document Information 3](#_Toc469918684)

[1.2 Intended audience 3](#_Toc469918685)

[1.2 References 3](#_Toc469918686)

[1.4 Glossary 3](#_Toc469918687)

[2. Introduction 4](#_Toc469918688)

[2.1 Background 4](#_Toc469918689)

[2.2 Purpose 4](#_Toc469918690)

[2.3 Scope 6](#_Toc469918691)

[2.4 Dependencies 6](#_Toc469918692)

[2.5 Assumptions 6](#_Toc469918693)

[3. Design Details 7](#_Toc469918694)

[3.1 Unblock my mobile 7](#_Toc469918695)

[3.2 End to End Data flow 8](#_Toc469918696)

[3.3 Application Architecture 8](#_Toc469918697)

[3.3.1 Sequence Diagram 9](#_Toc469918698)

[3.3.2 Class Diagram 10](#_Toc469918699)

[3.3.3 Deployment Diagram 11](#_Toc469918700)

[4. Detailed Design 12](#_Toc469918701)

[4.1 Business Logic 12](#_Toc469918702)

[4.2 Web Tier 13](#_Toc469918703)

[4.3 Configuration 13](#_Toc469918704)

[4.3.1 Application Level Parameters 13](#_Toc469918705)

[4.4 Packaging 13](#_Toc469918706)

[4.5 Security 14](#_Toc469918707)

[4.6 Transactions 14](#_Toc469918708)

[4.7 Logging Mechanism 15](#_Toc469918709)

[4.8 Testing 15](#_Toc469918710)

# Document Control

## 1.1 Document Information

|  |  |
| --- | --- |
| Title: | Design Document for GetO2PUK |
| Owner: | Sudhanva Mhaskar |
| For review by: |  |
| For approval by: |  |
| Distribution/Access: |  |
| Filename(s): | GetO2PUK\_Design\_Document\_V1.2.docx |

## 1.2 Intended audience

This document will be formally distributed to the following parties:

|  |  |
| --- | --- |
| Name | Company/Business Unit/Role |
| Phil Yarnall | BMV Project Manager |
| Russell Fairfield | Service Delivery Lead |

**Table 1 - Document distribution list**

## References

1. GetPUK Tech Ops Manual v1.0-1.doc

2. Detailed Design GetPuk v2.7-1.doc

3. GETO2PUK\_Requirement\_Document\_V0.1.docx

## 1.4 Glossary

|  |  |
| --- | --- |
| PUK | “Phone Unblock Keycode / PIN Unlock Key” = The Number used to unlock a mobile phone if it has been locked |
| MSISDN | A valid mobile number of an O2 customer. |
| Access Gateway | Gateway to call SOA service |
| SOA | Service Oriented Architecture |

# Introduction

## 2.1 Background

The GetO2PUK should allow a valid O2 Customer to get a PUK Code for their phone in case of Phone Lock and give detailed information to use that PUK with different handsets available.

## 2.2 Purpose

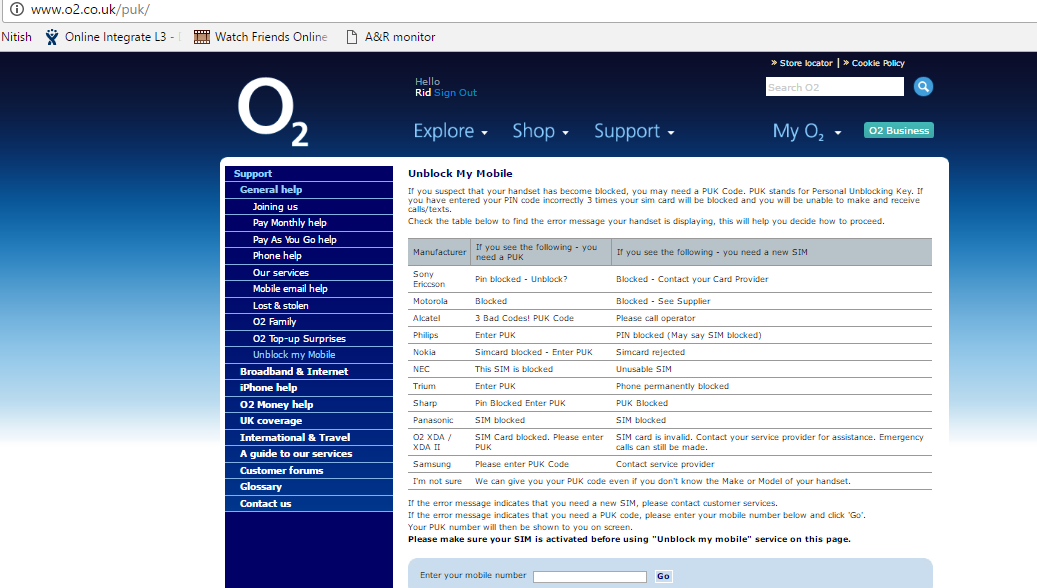
The purpose of this document is to provide documentation of the design and implementation.

Current PUK system are vulnerable to security threats and not meeting O2 standards.

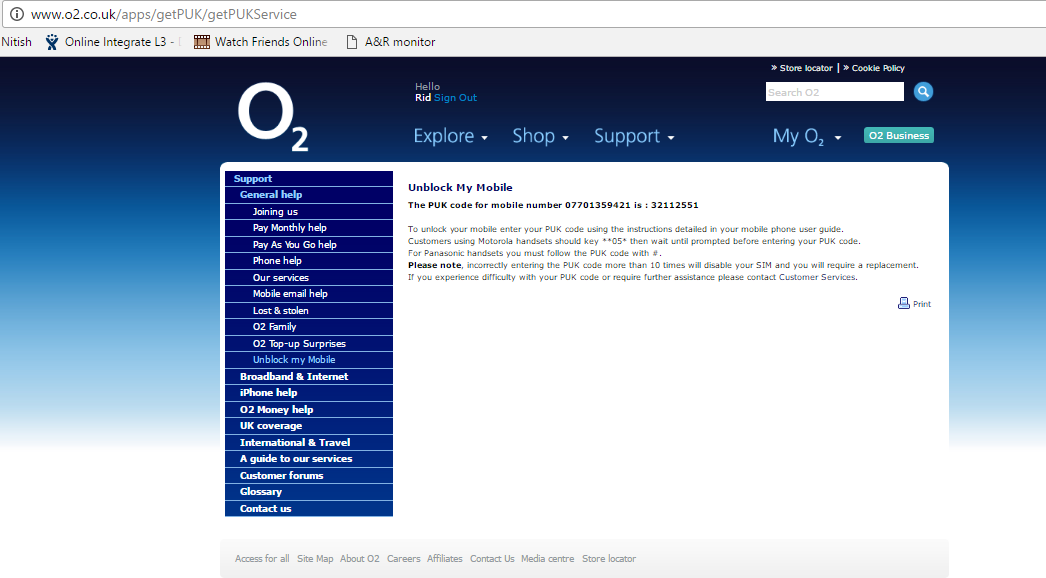
Main aim of redesigning this application is to overcome these which are explained in later section of this doc.

**Current Pages:**

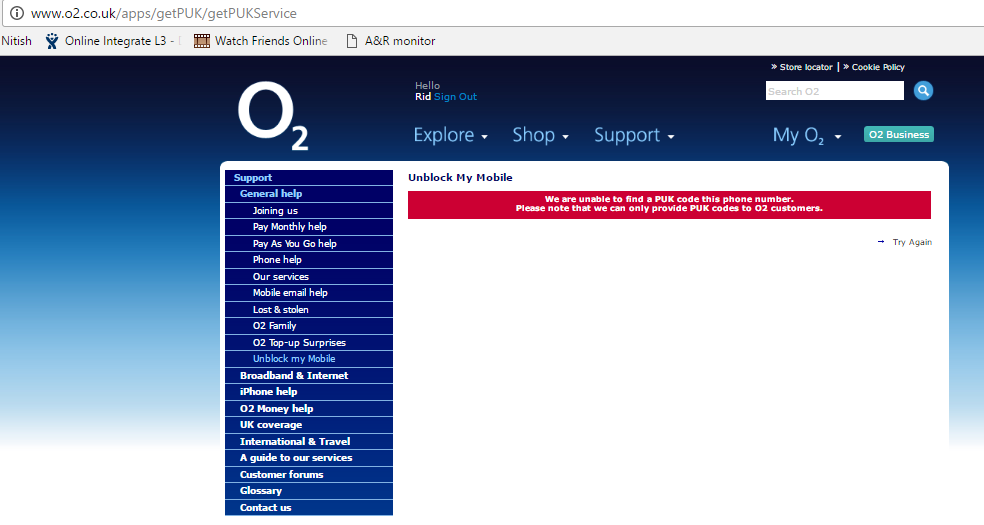
Welcome Page



Success Page



Failure Page



**New Pages**

Welcome Page

*TBC*

Success Page

*TBC*

Failure Page

*TBC*

## 2.3 Scope

This document covers the detailed design of the GETO2PUK Application.

## 2.4 Dependencies

This application will be highly dependent on Access gateway and SOA services.

## 2.5 Assumptions

The system will be deployed on the TOMCAT Application Server.

# Design Details

## 3.1 Unblock my mobile

* This operation provides A PUK which is required in Phone lock situation for an O2 Customer.
* For this, a valid O2 MSISDN is required which is blocked.
* For invalid MSISDNs, an appropriate failure message should be displayed.

Please refer to below High level data flow for GetO2PUK.

User visits link http://www.o2.co.uk/apps/getPUK/welcome

‘Unblock my mobile’ page is displayed

User enters MSISDN in field ‘Enter your mobile number’

Is the entered MSISDN a valid O2 number?

Yes

No

Appropriate failure message is displayed

Google reCaptcha checkpoint

Failure

Success

PUK for respective MSISDN is displayed

## 3.2 End to End Data flow

1. GetO2PUK Web Page
   1. The customer will request the Get PUK web page from [www.o2.co.uk](http://www.o2.co.uk) with /apps/getPUK/welcome in the path
2. Access Gateway
   1. App will call Access Gateway for captcha service and subscriber service.

AG does authentication and authorization for puk app

1. Backend System
   1. Google for verifying captcha details entered on welcome page and provide response back to Access gateway
   2. SOA which will call backend system to obtain puk code and provide response back to access gateway

1) GetO2PUK Web Page

2) Access Gateway

3.a) Google

3.b) SOA

## 3.3 Application Architecture

Google

TLS1.2 & SHA2

Web Tier

Access Gateway

Business Process Layer

GETO2PUK

App

External System /Interface Layer

GetO2PUK ErrorPage.vm

GetO2PUK SuccessPage.vm

GetO2PUK WelcomePage.vm

Presentation Layer

Initiation

Captcha verification

Captcha verification

O2

Customer

Get puk code

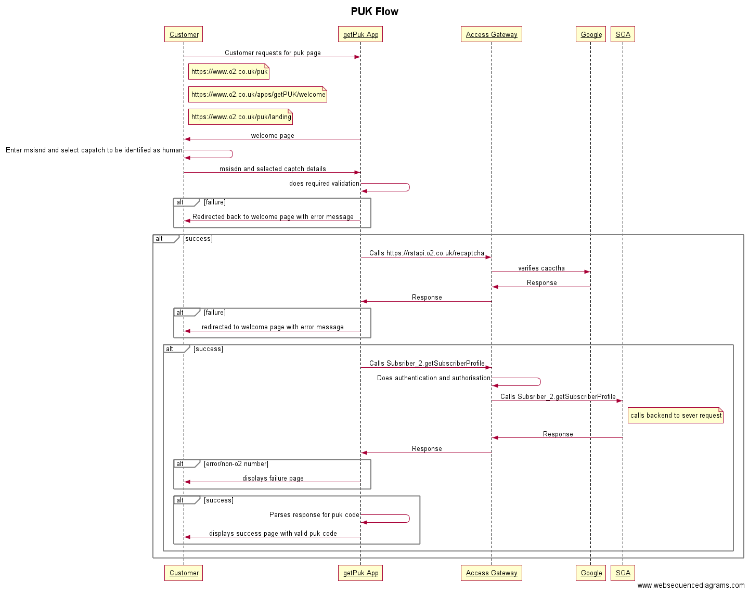
If error

Get puk code

SOA

If Success

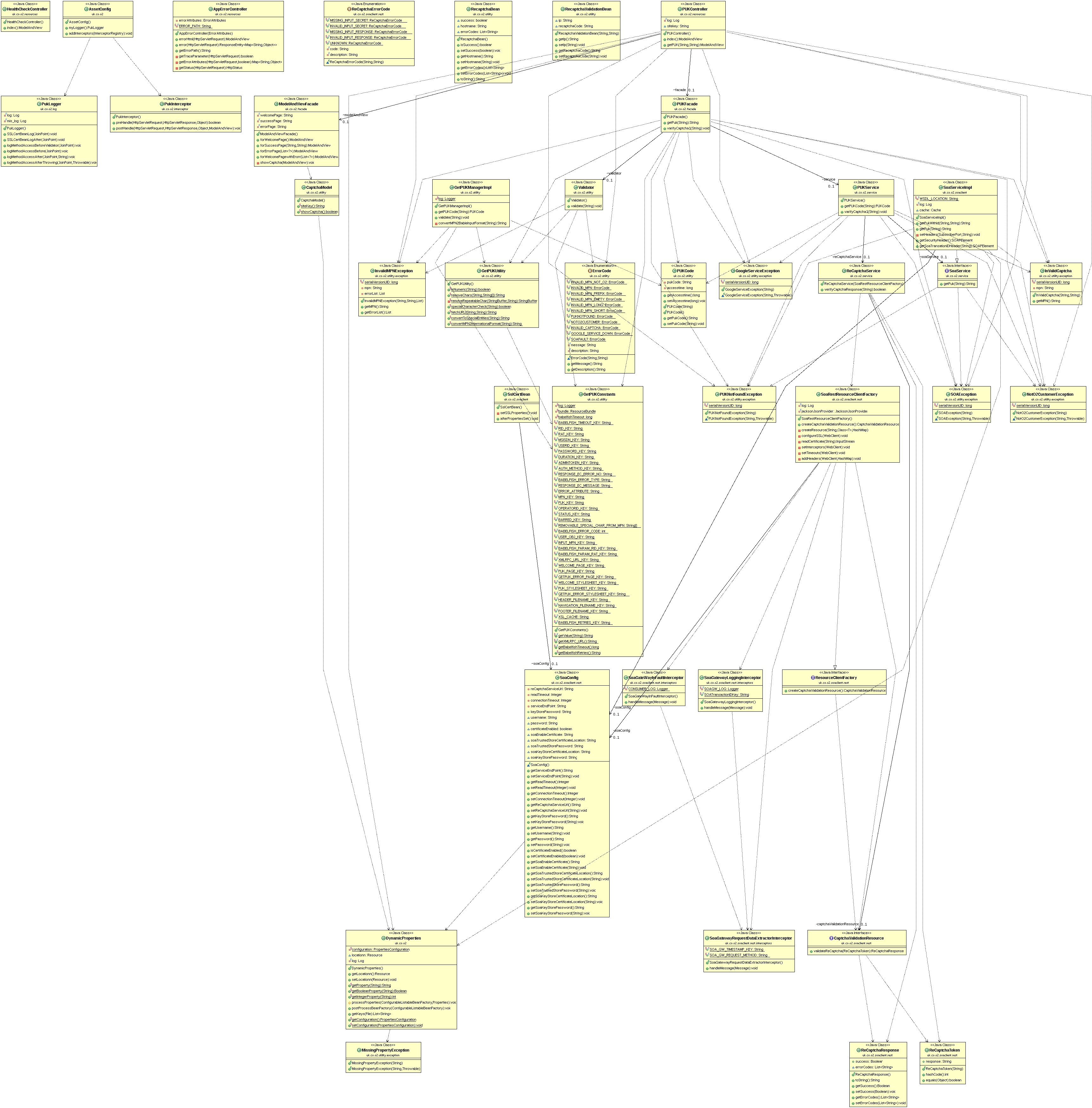
### 3.3.1 Sequence Diagram



***Figure 1 GetO2PUK Object Sequence diagram***

1. Customer visits o2 website and search for “PUK” / “Unblock my mobile”, it will land to welcome page
2. Customer enter captcha details and enters the MSISDN for which puk code is required
3. PUK application does required validation
4. If validation fails customer is redirected back to welcome page
5. Else, PUK application calls Access Gateway’s https://rstapi.o2.co.uk/recaptcha captcha service to verify human interaction
6. If Response returned from Access Gateway (google) has error, customer is redirected back to welcome page with proper error message
7. Else PUK app calls Access Gateway’s Subscriber\_2.getSubscriberProfile to get puk code
8. If there is any error from Access Gateway or if number is non-o2, customer is redirected to error page with valid error message
9. Else, customer is redirected to success page with PUK code displayed on the screen.

### 3.3.2 Class Diagram



### 3.3.3 Deployment Diagram

Application is deployed in AWS servers which hosts war file as well as content pages.

Content pages are kept outside of war file so it is easy to replace/modify as and when required.

****

# Detailed Design

## 4.1 Business Logic

The fundamental business logic behind the GETO2PUK application is to fetch the PUK code for a user specified MSISDN. The O2 User by selecting the ‘Unblock my mobile’ link; is directed to GetO2PUK welcome page.

This MSISDN is validated for below rules,

1. It should be numeric value except '+' symbol at the beginning or 1st character.

2. If it starts with 0 it should be mandatory followed by 7; i.e. 07 should be the first two numerals and total length should be 11 digits.

3. If it starts with 44 it should have a total length of 12 digits.

4. If it starts with +44 it should have a total length of 13 digits.

Numbers starting with 0044 are not supported.

Once these 4 cardinalities are satisfied. The GETO2PUK web component invokes the Access gateway communicator for captcha verification and if captcha is verified successfully application again makes call to Access Gateway with the requested MSISDN. Here it makes SOA call to find the PUK code for specific MSISDN. The returned result set is verified for any erroneous condition by checking the error code and error message. If there isn’t any error code then it presumed that PUK code fetch is successful and obtain the PUK code from the Result page.

In case if any erroneous scenario occurs like captcha verification or fetching the PUK code; GETO2PUK Error page is displayed with appropriate error message. On the similar lines if there is any MSISDN validation error occurred, message will be displayed on the GETO2PUK Welcome with red marked appropriate error message.

Application is capable of caching data in memory which can be enabled using flag available in application.property file.

Apart from this flag there is one more flag for enabling/disabling google captcha service to avoid any impact on customer in case of captcha service is down.

Also there would be unique transaction id generated and logged for each request.

## 4.2 Web Tier

The application web tier comprises of presentation layer and business logic/service layer. The presentation layer does contain the velocity templates pertaining to welcome page, error page and success page and business logic layer contains spring controllers, Services, Access Gateway communicator, GetO2PUK Manager Implementation and supporting utility/value-object classes.

## 4.3 Configuration

There are different parameters need to be configured varying from application level configuration to logging configuration parameters within properties files.

Logic for properties file load:

1. Application should read property file from external location on the server.
2. If property file is not present at external location, then property file which is packaged in war would be read by the application.
3. If files are present at both locations then what should happen is explained below:
   1. If external property file does not contains all the propertied (keys) from packaged (war) property file, then deployment should fail with proper exception.
   2. If external property contains extra keys then they should be logged and application will start smoothly.

### 4.3.1 Application Level Parameters

The application level parameters include Access Gateway parameters, UI pages and content path.

## 4.4 Packaging

Gradle build tool is used to compile and build this application. Precisely running “build” task.

getPuk.war is built using the gradle build tool and the build.gradle build file.

The properties file is selected depending on the deployment environment dev, ref or live. Each environment has its own properties file.

## 4.5 Security

Current getpuk application has following security issues:

1. It connects to outside system (bablefish) over http i.e. no ssl
2. Technologies being used are old and hackable
3. Application is vulnerable to DDOS attacks

To overcome these:

1. New application is being developed issues via latest technology stack i.e spring and java8 etc.
2. Captcha service is implemented to make sure only human can access the service and robotic actions are prevented
3. Communication to outside world (Access gateway Communication) is secure as per latest security standards i.e. on SHA2 & TLS 1.2.

## 4.6 Transactions

**SOA REQUEST EXAMPLE:**

---[HTTP request - http://localhost:8091/Service/Subscriber\_2\_0]---

Accept: text/xml, multipart/related

Content-Type: text/xml; charset=utf-8

SOAPAction: ""

User-Agent: JAX-WS RI 2.2.10 svn-revision#919b322c92f13ad085a933e8dd6dd35d494736

4b

<?xml version='1.0' encoding='UTF-8'?><S:Envelope xmlns:S="http://schemas.xmlsoa

p.org/soap/envelope/"><S:Header><cor:SOAConsumerTransactionID xmlns:cor="http://

soa.o2.co.uk/coredata\_1.xsd">8a2cb3f3-3824-41de-b4ef-7c440e15ce5d:puk</cor:SOACo

nsumerTransactionID><wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/20

04/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"><wsse:UsernameToken xmlns:wsse

="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.

xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecuri

ty-utility-1.0.xsd"><wsse:Username>online\_puk\_2457</wsse:Username><wsse:Password

Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-pr

ofile-1.0#PasswordText">sIoTsbdv</wsse:Password></wsse:UsernameToken></wsse:Secu

rity></S:Header><S:Body><ns2:getSubscriberProfile xmlns="http://soa.o2.co.uk/cor

edata\_1" xmlns:ns2="http://soa.o2.co.uk/subscriberdata\_2"><ns2:subscriberID>4477

04610260</ns2:subscriberID></ns2:getSubscriberProfile></S:Body></S:Envelope>----

----------------

**SOA RESPONSE EXAMPLE:**

---[HTTP response - http://localhost:8091/Service/Subscriber\_2\_0 - 200]---

null: HTTP/1.1 200

Content-Length: 335

Content-Type: text/xml;charset=UTF-8

Date: Fri, 02 Dec 2016 13:25:24 GMT

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Body

><getSubscriberProfileResponse xmlns="http://soa.o2.co.uk/subscriberdata\_2" xmln

s:ns2="http://soa.o2.co.uk/coredata\_1"><subscriberProfile><operator>O2</operator

><puk>9989</puk></subscriberProfile></getSubscriberProfileResponse></soap:Body><

/soap:Envelope>--------------------

## 4.7 Logging Mechanism

The Application Level runtime logging is registered for business components either on to server console or on to log file or on to both.

Below logs files would be present on server:

*getPUK2.log*: Holds information about validation, request, response and exception information along with stack trace

*getPUK2mis.log*: Holds information about mpn and corresponding puk

*console.log*: Holds information about server start/stop and other server related details

## 4.8 Testing

Mockito will be used for unit testing, and Selenium for automated end to end testing.