Product Registration

Interface specification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document History** | | | | |
| **Version** | **Date** | **Author** | **Section** | **Changes** |
| 0.1 | 08-01-2016 | Viswaradh Reddy A | All | Initial draft |
| 0.2 | 23-03-2016 | Kiran Kumar R | All | Initial draft |

|  |  |
| --- | --- |
| **Author** | Kiran Kumar R |
| **Approved by** |  |
| **Email Id** | [Kiran.kumar.r\_1@philips.com](mailto:Kiran.kumar.r_1@philips.com) |

CONTENTS

1. DEFINITIONS & ABBREVATIONS 3

2. INTRODUCTION 3

2.1 Purpose 3

2.2 Scope 3

2.3 Target Audience 3

2.4 References 3

3. ARCHITECTURAL ANALYSIS 4

3.1 Product Overview 4

3.2 System Context 4

3.3 Future Extensions 5

3.4 Design Method 5

3.5 Platform dependency 5

3.5.1 Class names 5

3.5.2 Errors 5

4. STATIC DESIGN 7

4.1 Overall System Architecture 7

4.2 Layering Model 7

4.2.1 Product Registration Interface

4.2.2 PRX backend 12

5. DYNAMIC DESIGN 13

5.1 Use cases 13

5.1.1 Use case: 13

6. Appendix 13

6.1 Configuration 13

# DEFINITIONS & ABBREVATIONS

|  |  |
| --- | --- |
| API | Application Programming Interface |
| CDP | Connected Digital Propositions |
| HTTP | Hyper Text Transfer Protocol |
| RCDH |  |
| UUID | Universal Unique Identifier |
| PRX |  |

* Product registration – The process where register a product or register multiple products and fetching the registered products w.r.t a user account.
* PRX – Platform that provides the interface to retrieve product information and also provides interface w.r.t product registration.
* Janrain – Platform that provides service interface.

# INTRODUCTION

This document provides an overview of architecture and design for product registration feature in Mobile applications.

## Purpose

The purpose of this document is to elaborate various building blocks used in product registration of Philips products using mobile applications. This document also explains how these building blocks can be put together to build a re-usable component.

## Scope

This document covers only product registration from mobile touch point. Also, the design does not cover the data sync mechanism between various backend components.

The component design is not specific to any particular platform though iOS and Android will be the first carrier platforms.

## Target Audience

CDP Development teams

## References

* Product registration in mobile apps\_analysis 23-9-2015 pptx
* Product registration - Design specification-0.3\_art direction required.pdf
* UX flow\_ product registration\_v0.8.pdf
* PRX\_API\_v4.18.docx

# ARCHITECTURAL ANALYSIS

## Product Overview

Product registration is one the main features of the mobile applications developed in Philips. Philips earlier used email/password registration mechanism to register the user and link the product to the user. As one of the step to ease the process of registration, Social media login using Janrain is being integrated.

As part of Digital Innovation, Product registration is a horizontal component that can be reused across various applications. The subsequent sections provide the detail of the product registration as a re-usable component.

## System Context

The following diagram shows interaction of Mobile application with various systems. The document concentrates on the direct touch points of mobile application.

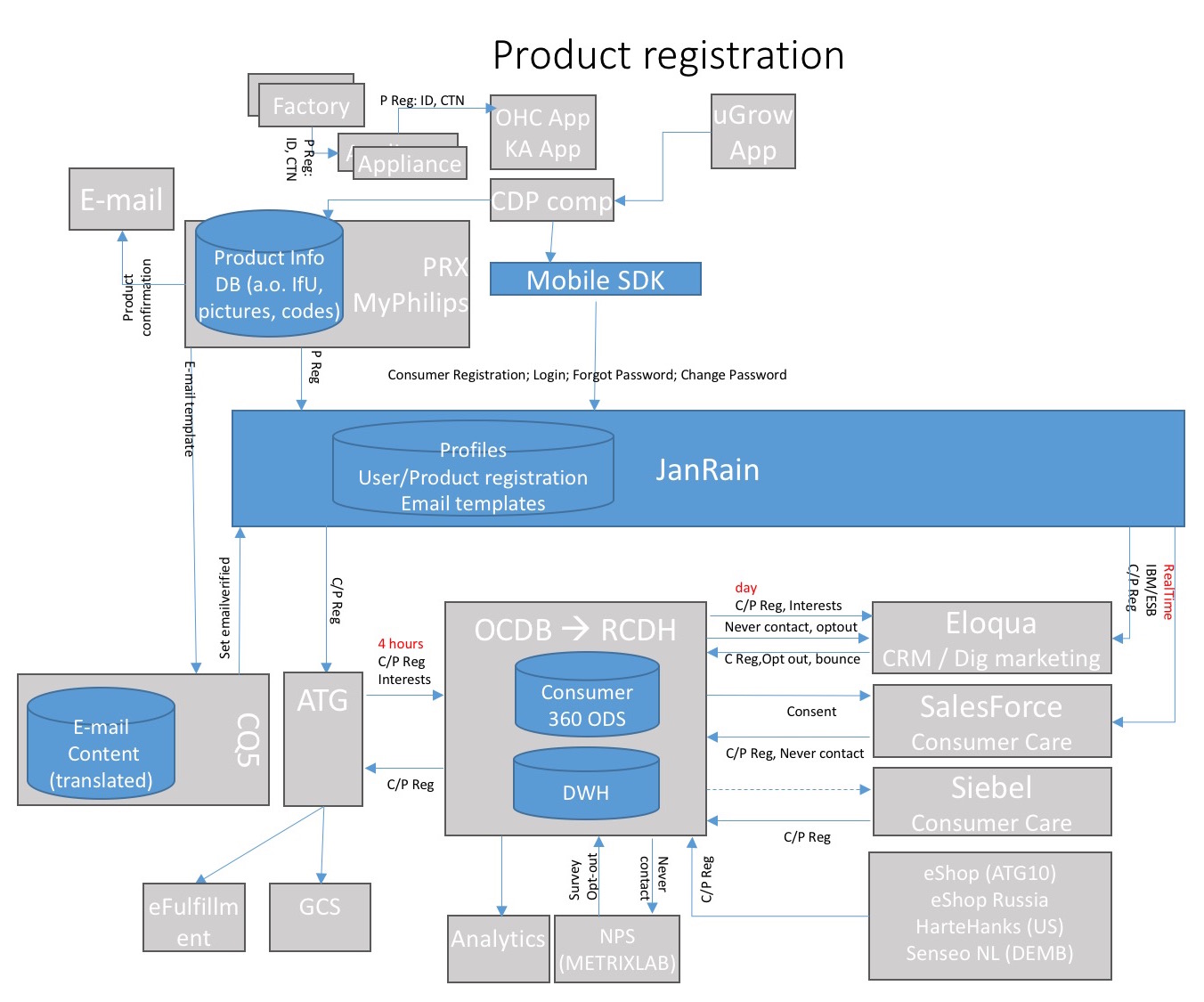


Figure 1: System context

## Future Extensions

* To able to search the product deepened on the catalog and register the product.

## Design Method

The design used Object Oriented concepts which are supported by majority of the platforms like Objective C and Java. The design patterns like Façade, Delegation, and Extensions are used.

All communication with server (PRX) will be using SSL certificates. Plain HTTP communication will not be used.

The Registration Component will not expose any internal APIs and classes. This is to avoid applications bypassing the Registration component and directly invoking internal functions thus compromising the rationale behind horizontal component.

The components follow the layered approach. Following rules are enforced for how the layers interact with each other.

* **One way interaction (Top to down)**: Higher level layers can interact with layers below, but lower layers cannot interact with layers above. This is to avoid circular dependencies between layers.
* **Strict interaction**: Layers can only interact with layers below. This will make sure that modifications in one layer will only affect layers above.

## Platform dependency

While Object Oriented concepts are used in iOS and Android, the implementation details differ. The following sections provide a brief overview of such implementation details.

### Class names

The class names used here can be modified based on the platform. In iOS, class names are prefixed with product names (Ex: User class is implemented as PPRRequestHandler. PPR Indicates PhilipsProductRegistration). In Android it is prefixed with package names like com.philips.cdp.ppr.registrableProduct.

### Errors

iOS uses NSError to indicate errors while Android uses exceptions. The design here provides an abstract of error and not the implementation details.

# STATIC DESIGN

## Overall System Architecture

The following diagram shows the over system architecture of user and product registration using PRX and Janrain.

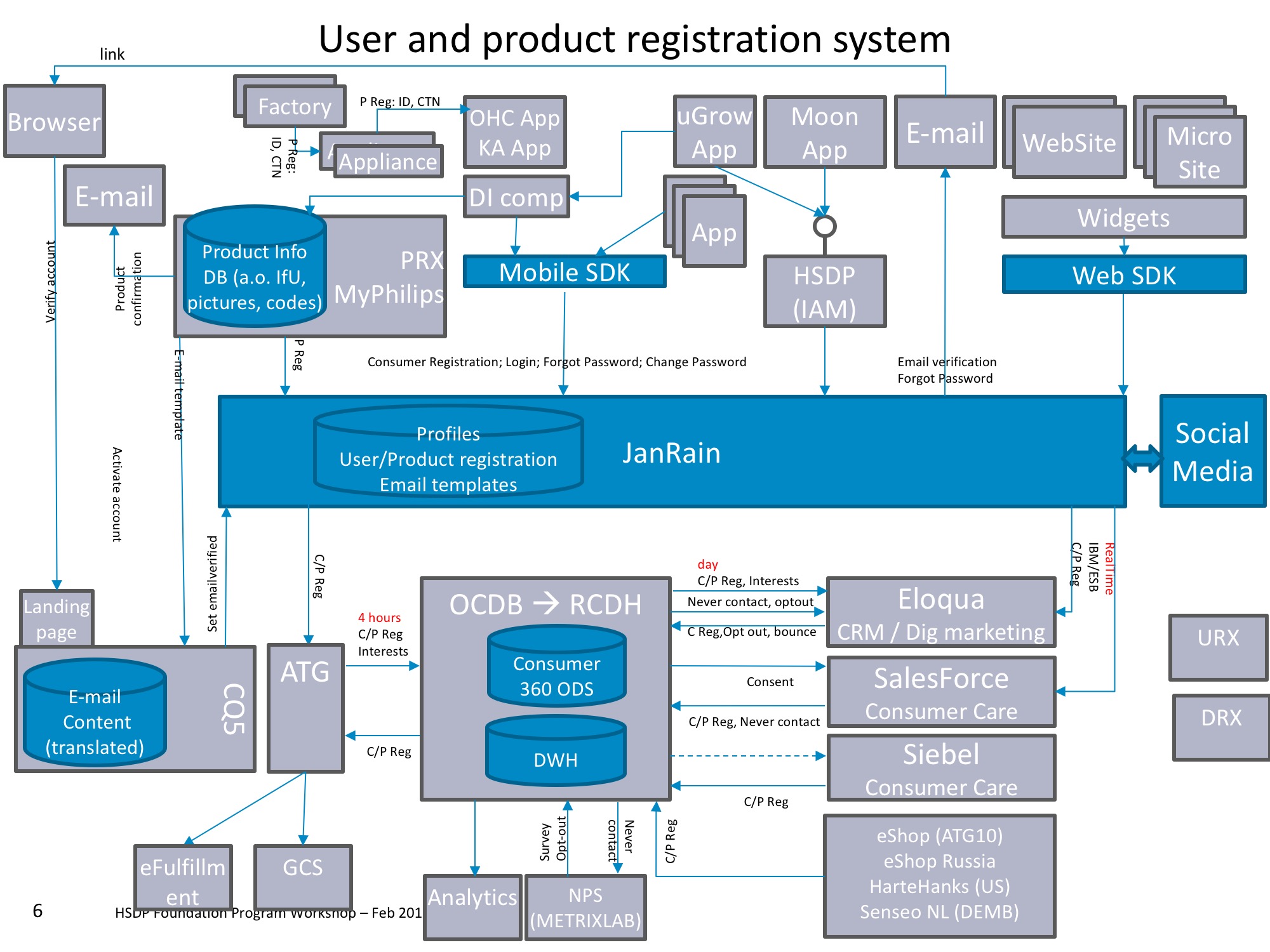


Figure 2: Registration system architecture

Ref: Product registration in mobile apps\_analysis 23-9-2015 pptx

The diagram explains over all system overview for user and product registration.

The mobile application interacts with various systems in order to perform user and product registration. The focus of this document is to “unify” all these components (PRX, Janrain SDK, backend API) into a re-usable mobile component that can be used across various mobile applications. The following diagram depicts the same.

## Layering Model

### Product Registration Interface

### Registration Interface – Product (Just for information – currently its part of user registration)

This class exposes product navigation and registration functionalities. It interacts with PRX server to realize the functionalities. For information on PRX APIs, refer “PRX\_API\_v4.18.docx”.

#### Responsibilities/Functions

Following functions are provided by this layer

* Product registration
* Fetching product metadata
* Retrieve registered products for a user.

#### Class diagram

Following figure shows the class diagram for the ProductRegistration.

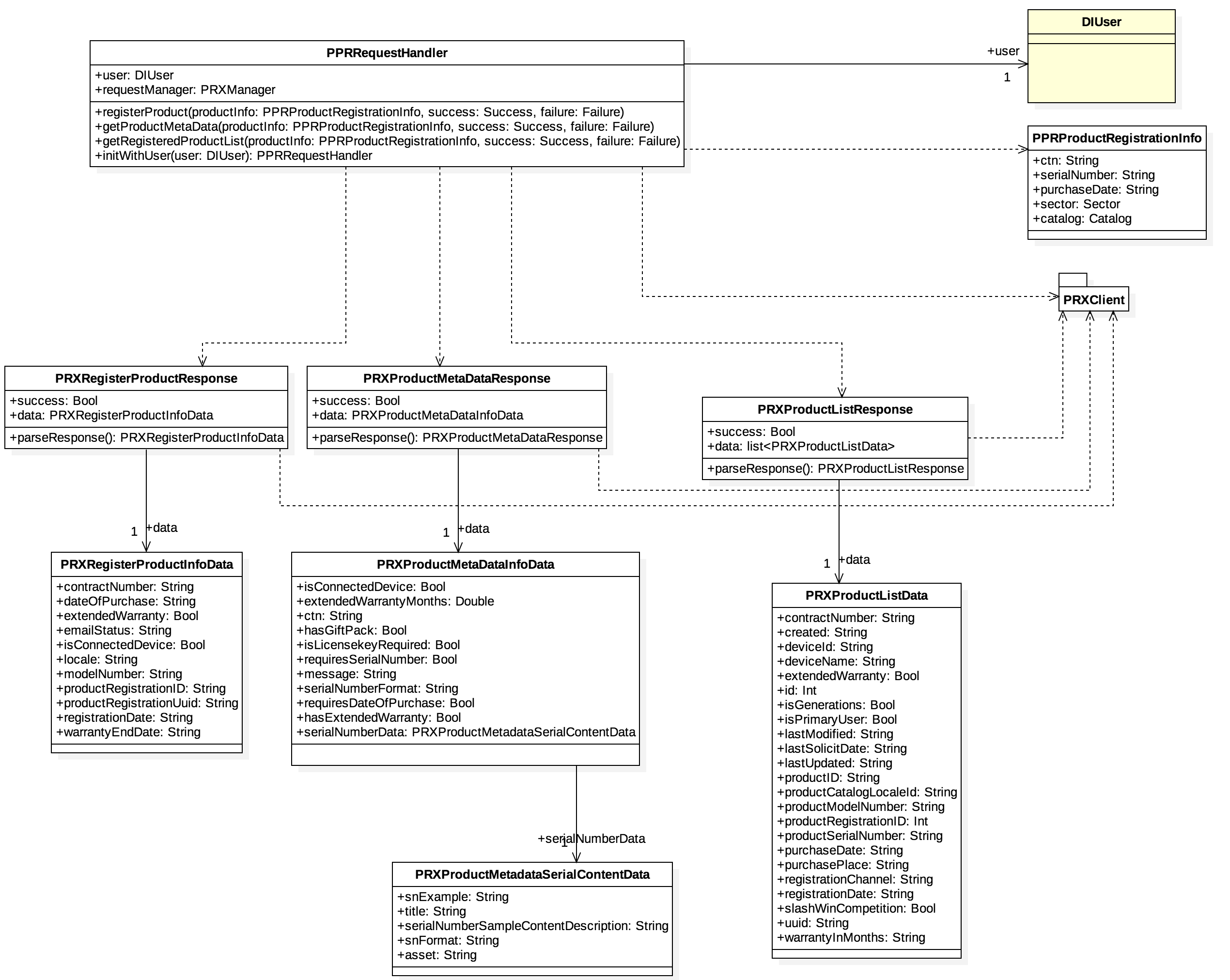


Figure 3: Product Registration

The diagram below shows how the app registers product using PRX APIs. Mobile app makes a call to PRX server to Register product using access token.

*(Picture reference: C020\_4-Identity\_Access Management-Product registration.pptx)*

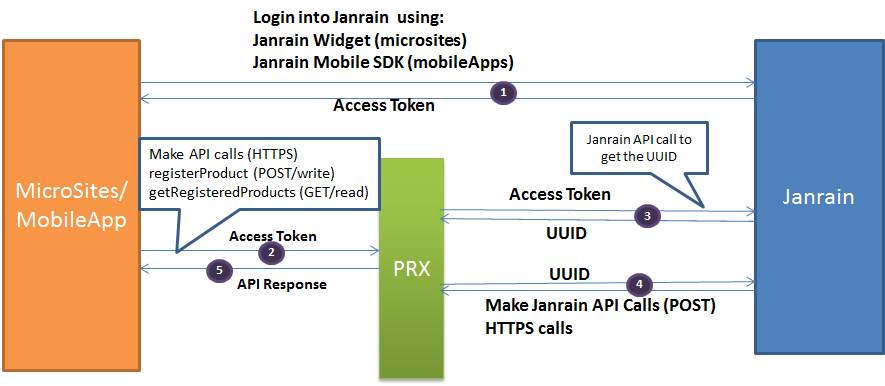


Figure 4: Usage of Philips service for product registration

#### Interfaces

##### register

|  |  |
| --- | --- |
| Function definition | registerProduct(ProductRegistrationInfo productInfo,Success success,Failure failure) |
| Parameters | productInfo: Information of the product to be registered |
| Pre-Condition | User must be logged in, CTN is must |
| Brief | This function calls PRX API to register the product with the user. It does the following actions:   * Call PRX API to register product by providing productInfo and access token   The status of the operation is reported to application through delegate/handler. |
| Error scenarios | * Invalid parameter * Network errors * System errors |
| Callback functions | * Success – Successfully registered the product. * Failure – Product could not be registered. |
| Type | Asynchronous |

##### getRegisteredProducts

|  |  |
| --- | --- |
| Function definition | getRegisteredProdcutList(ProductRegistrationInfo productInfo,Success success,Failure failure) |
| Parameters | User must be logged in |
| Pre-Condition | None |
| Brief | This function calls PRX API to fetch the list of registered products for the user. It does the following actions:   * Call PRX API to retrieve registered products by providing access token of the user.   The status of the operation is reported to application through delegate/handler. If successful, the list of products (ProductRegistrationInfo) is provided in the delegate/handler. |
| Error scenarios | * Invalid parameter * Network errors |
| Callback functions | * Success – Successfully fetched the list of registered products. * Failure – Registered products could not be fetched. |
| Type | Asynchronous |

##### getMetadata

|  |  |
| --- | --- |
| Function definition | getProductMetadata(ProductRegistrationInfo productInfo,Success success,Failure failure) |
| Parameters | productInfo: Information of the product to be registered |
| Pre-Condition | CTN is must |
| Brief | This function calls PRX API to register the product with the user. It does the following actions:   * Call PRX API to register product by providing productInfo and access token   The status of the operation is reported to application through delegate/handler. |
| Error scenarios | * Invalid parameter * Network errors * System errors |
| Callback functions | * Success – Successfully registered the product. * Failure – Product could not be registered. |
| Type | Asynchronous |

### PRX backend

PRX backend component provides the product registration features. This component contains the APIs to interact with REST interfaces.

The following functions are provided by Janrain backend.

* Register the product
* Fetch the registered products w.r.t user
* Fetch the product summary
* Fetch the product

# Dynamic Design

## Use cases

### Use case:

# Appendix

## Configuration

Product registration is depend on the user registration component, to register a product , the user must login.

There are some application specific parameters that are configured by application. These are:

* CTN of products need to configured w.r.t locale