Product-Service-Resource Information Model E-mail, a PLM Case Study

Release 1.1

TR153

Version 1.3



July, 2010

Notice

This document has been through review cycles; however, due to the inherent complexity in the design and implementation of software and systems, no liability is accepted for any errors or omissions or for consequences of any use made of this document.

Under no circumstances will the TM Forum be liable for direct or indirect damages or any costs or losses resulting from the use of this specification. The risk of designing and implementing products in accordance with this specification is borne solely by the user of this specification.

This document is a copyrighted document of TM Forum and its use by members and non-members of TM Forum is governed by all of the terms and conditions of the Agreement on Intellectual Property Rights between TM Forum and its members (<http://www.tmforum.org/IPRAgreement/2211/home.html>) and may involve a claim of patent rights by one or more TM Forum members or by non-members of TM Forum.

Direct inquiries to the TM Forum office:

240 Headquarters Plaza,

East Tower – 10th Floor,

Morristown, NJ  07960 USA

Tel No.  +1 973 944 5100

Fax No.  +1 973 944 5110

TM Forum Web Page: [www.tmforum.org](http://www.tmforum.org/)

Table of Contents

[Notice 2](#_Toc266784472)

[Table of Contents 3](#_Toc266784473)

[Executive Summary 5](#_Toc266784474)

[1 Introduction 6](#_Toc266784475)

[2 An Example Case Study 10](#_Toc266784476)

[2.1 The Product Marketing Plan 10](#_Toc266784477)

[2.1.1 Domain-specific Analysis 10](#_Toc266784478)

[2.1.2 SID Model 14](#_Toc266784479)

[2.1.2.1 Product catalog for user-consumes-services approach 15](#_Toc266784480)

[2.1.2.2 Product catalog for user-consumes-product approach 16](#_Toc266784481)

[2.1.2.3 Modeling considerations with regard to product offerings 19](#_Toc266784482)

[2.1.3 Requirements to Lower Layers 21](#_Toc266784483)

[2.2 Sellable Products 21](#_Toc266784484)

[2.2.1 Domain-specific Analysis 21](#_Toc266784485)

[2.2.2 SID Model 22](#_Toc266784486)

[2.2.3 Requirements to Lower Layers 26](#_Toc266784487)

[2.3 Customer-facing Services 27](#_Toc266784488)

[2.3.1 Domain-specific Analysis 27](#_Toc266784489)

[2.3.1.1 Modeling Systems and Services with UML 30](#_Toc266784490)

[2.3.1.2 Business Entities that complement Customer-facing Services 33](#_Toc266784491)

[2.3.2 SID Model 34](#_Toc266784492)

[2.3.3 Requirements to Lower Layers 37](#_Toc266784493)

[2.4 Linking Products with Customer-facing Services 37](#_Toc266784494)

[2.4.1 Domain-specific Analysis 37](#_Toc266784495)

[2.4.2 SID Model 37](#_Toc266784496)

[2.5 Resource-facing Services 40](#_Toc266784497)

[2.5.1 Domain-specific Analysis 40](#_Toc266784498)

[2.5.2 SID Model 42](#_Toc266784499)

[2.5.3 Requirements to Lower Layers 44](#_Toc266784500)

[2.6 Linking Customer- and Resource-facing Services 45](#_Toc266784501)

[2.6.1 Domain-specific Analysis 45](#_Toc266784502)

[2.6.2 SID Model 46](#_Toc266784503)

[2.7 Logical Resources of the Infrastructure 47](#_Toc266784504)

[2.7.1 Domain-specific Analysis 47](#_Toc266784505)

[2.7.2 SID Model 49](#_Toc266784506)

[2.7.3 Requirements to Lower Layers 52](#_Toc266784507)

[2.8 Production of Resource-facing Services by Logical Resources 52](#_Toc266784508)

[2.8.1 Domain-specific Analysis 52](#_Toc266784509)

[2.8.2 SID Model 53](#_Toc266784510)

[2.9 Physical Resources 55](#_Toc266784511)

[2.9.1 Domain-specific Analysis 55](#_Toc266784512)

[2.9.2 SID Model 56](#_Toc266784513)

[2.10 Linking Logical and Physical Resources 58](#_Toc266784514)

[2.10.1 Domain-specific Analysis 58](#_Toc266784515)

[2.10.2 SID Model 58](#_Toc266784516)

[3 Summary 60](#_Toc266784517)

[4 Appendix 1 – Glossary 61](#_Toc266784518)

[5 Appendix 2 – The Relationship between Customers, Products and Customer-facing Services 67](#_Toc266784519)

[6 Administrative Appendix 69](#_Toc266784520)

[6.1 Version History 69](#_Toc266784521)

[6.2 Company Contact Details 69](#_Toc266784522)

[6.3 Acknowledgments 70](#_Toc266784523)

Executive Summary

The case study presents the results of an extensive modelling exercise for an example offering of e-mail undertaken by the TMF Product Lifecycle Management (PLM) Team. Analysis techniques from systems engineering like use cases and component diagrams are combined with key concepts of the TMF Information Framework (aka Shared Information Data Model - SID) to decompose an e-mail offering into a landscape of building blocks that can be mapped into product, service and resource catalogues and managed in lifecycle processes.

# Introduction

|  |
| --- |
| *Essentially, all models are wrong, but some are useful.*  *G. Box* |

This case study presents a top-down analysis of an example email service from the product marketing level down to the resources. It identifies the different levels of the construction of a marketable product from services and resources and explains the major business entities in a tutorial style.

The goal of the case study is to build a common understanding of these major entities and their relations on the basis of a real-world example.

The authors are convinced that it is a necessary step to first understand the semantics of the building blocks that are to be managed in a PLM lifecycle before analyzing their lifecycle properties, in other words: We need to answer the question: “What are the essential parts of a product/service construction?” before answering the question “How do we manage the parts?”

The case study uses decomposition levels for the analysis of product construction. Each decomposition level reflects a unique viewpoint. While doing the top-down analysis (product to service to resource), each level is the source of requirements to the level below. Different decomposition levels are shown in the table 1 and the corresponding viewpoint.

The demarcation lines between products, customer- and resource-facing services have been the object of many heated and frustrating discussions. From discussions about the example presented in this case study, it became obvious that there are at least two approaches within the TMF community about the relationship of customers and users to products and customer-facing services.

* The user-consumes-service approach. In this modeling approach, customer and provider agree commercially in terms of products, while users consume services, e.g. when reading mails or making phone calls
* The user-consumes-product approach. In this modeling approach, customers and users always interact with products, products are like a façade towards them and services are only created in the provider’s infrastructure

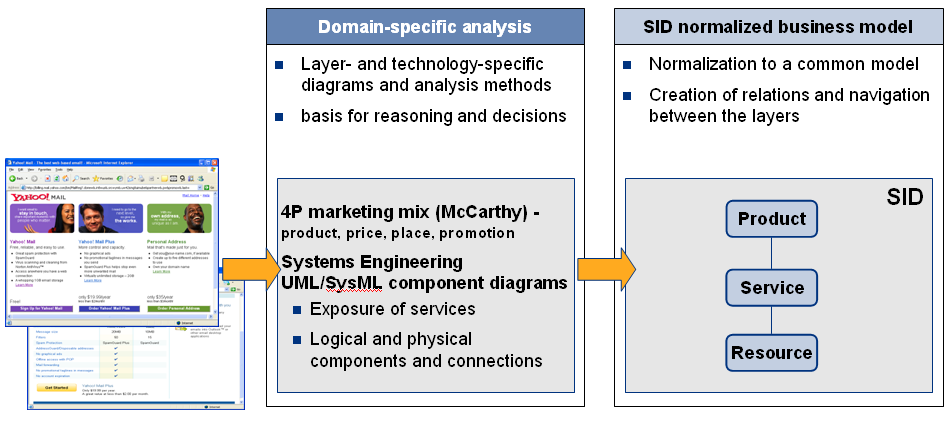
The two modeling approaches and their key differences are explained in Appendix 2. This case study was originally created on basis of the user-consumes-service modeling approach. In the current version, some illustrations and discussions regarding the user-consumes-product modeling approach have been added.

In order to make clear which modeling approach is presented in a section of this document, the applicable sections discussing the user-consumes-product approach are marked by a vertical line.

|  |  |  |  |
| --- | --- | --- | --- |
| **eTOM Level** | **Decomposition Level** | **Viewpoint** | |
| Product | Product Marketing Plan | Market focus  What products shall the provider’s portfolio sell at which prices over which channels promoted by which campaigns, in order to maximize profit and shareholder value according to the marketing strategy?  ⇨ Source of requirements for the use, creation, change or disposal of a sellable product | |
| Sellable Products | *User-consumes-service approach:*  Customer focus (buyer role)  What is the detailed definition of the product so that it can be offered, ordered and billed, in order to implement the marketing plan? | *User-consumes-product approach:*  Customer focus (buyer and user role)  What is the detailed definition of the product so that it can be offered, ordered and billed, in order to implement the marketing plan?  What is the functionality that a user can access to obtain the value that is promised by a product? |
| ⇨ Source of requirements for the use, creation, change or disposal of customer-facing services | |
| Service | Customer-facing Services | *User-consumes-service approach:*  Customer focus (user role)  What is the functionality that a user can access to obtain the value that is promised by a product? | *User-consumes-product approach:*  Infrastructure functionality focus  What is the functionality that the service provider infrastructure needs to provide, without consid­ering a particular implementation? |
| ⇨ Source of requirements for the use, creation, change or disposal of resource-facing services | |
| Resource-facing Services | Equipment interface focus  How are messages exchanged between external equipment and provider infrastructure in order to make the customer-facing services available to users?  ⇨ Source of requirements for the use, creation, change or disposal of logical resources | |
| Resource | Logical Resources | Software focus  What software and other logical entities implement the resource-facing services?  ⇨ Source of requirements for the use, creation, change or disposal of physical resources | |
| Physical Resources | Hardware focus  What hardware and other physical entities provide the runtime-environment for the logical resources? | |
| Supplier/ Partner |  | - out of scope - | |

Table : Decomposition levels used in the case study

The case study begins with the description of each decomposition level with a domain-specific analysis, followed by a SID normalized business model, as shown in Figure 1. This is closely reflected in the structure of the document. The domain-specific analysis uses diagrams and analysis methods that are familiar to the domain experts, e.g. product managers or application engineers. The SID is then used as a normalized business model that over spans all decomposition levels and creates the relations between the layers.

Figure : Domain-specific analysis and SID normalized business model

The top-down approach taken for the identification of the entities to be managed on each layer does not imply that the case-study proposes a top-down waterfall style of a product construction process. In the contrary, it is the vision of the PLM team to make the capabilities that are present in a layer manageable by handling them as entities with a lifecycle. An ideal product creation scenario would not ripple down from the marketing plan to the physical resources, but instead reuse as many of the existing entities as possible.

The case study intends to give the concepts of the SID a real-world meaning, sometimes these interpretations go beyond the definitions and hints given in the SID and eTOM primary documentation.

The glossary in the appendix summarizes the definitions from the standards and complements them with the pragmatic viewpoint of this case study.

The case study proposes a practical approach that builds bridges between systems engineering, UML component modeling, service modeling, SID and eTOM.

For that purpose, the case study applies ideas and concepts of following reference models:

* 4P marketing mix: The concept of the 4P marketing mix (product, place, price, promotion) has been proposed by Jerome McCarthy in 1960 and is the basis of many marketing books and classes
* Systems engineering: The case study applies systems engineering methodology for the analysis of customer- and resource-facing services. The fundamental idea is to define a system boundary and then to identify the interaction with stakeholders external to the boundary. For that purpose, a definition of the term “system” is taken from the INCOSE systems engineering handbook
* UML/SysML: The case study uses UML component models in order to create business models of systems that expose services at ports over a prescribed interface. The component modeling is influenced by SysML, a UML profile promoted by the OMG for systems engineering
* OASIS: The case study keeps its interpretation of the concept “service” in line with the definition of OASIS
* TMF eTOM: The case study performs a top-down decomposition of the mail example according to the eTOM layering of product, service and resource. The case study focuses on the entities that are involved in the product construction, the particular activities of the eTOM are not in scope
* TMF SID: The case study explains how the SID can be used to build a high-level conceptual model of a product and its underlying services and resources

# An Example Case Study

Starting on the basis of a real service which offering has been taken from its internet Web page, this section elaborates a model of the Product, Service and Resource levels. The focus of the model is to build a common understanding of the composition of a product from underlying services and resources, intentionally ignoring the procedural aspects of designing, implementing, provisioning and operating.

While the service presented, Yahoo Mail, is taken from the real world, the content of this section is the result of several discussions between the members of the team and does not necessary reflect the actual implementation by the corresponding service provider, Yahoo in this case.

This example has been selected, among others, for the following reasons:

* it is a real one
* the way it is presented (its offering) looks simple, so we can imagine it could be simple to model
* mail service in general is a very common and used by almost every internet user on a regular basis

## The Product Marketing Plan

### Domain-specific Analysis

The product marketing plan determines the key tactical elements that implement a service provider’s marketing strategy. The elements are referred to as the four P’s of marketing[[1]](#footnote-1) (marketing mix): Product, Price, Place and Promotion.

4P Product. In marketing terms the 4P Product refers to tangible, physical products as well as services. This element of the marketing plan reflects the intentional product decisions made by the service provider; these include the value proposition, the functionality, the quality, the branding and all other aspects that are known to be important to the customers in a targeted market segment.

4P Price. The Price element of the marketing plan defines the pricing policy applied by the service provider, e.g. volume based pricing vs. flat rate, subsidies for handsets, volume discounts and wholesale pricing.

4P Place. The Place element represents the distribution decisions that are taken by the service provider, e.g. the sales channels used to sell the products and the market segments that are addressed.

4P Promotion. The Promotion element represents the various aspects of communicating about the product with the goal of generation a positive customer response (a “lead”).

In total, the marketing plan determines how the service provider addresses the market with the right products at competitive prices over effective channels and well-targeted promotion, all in order to increase shareholder value and get a maximum return on its investments.

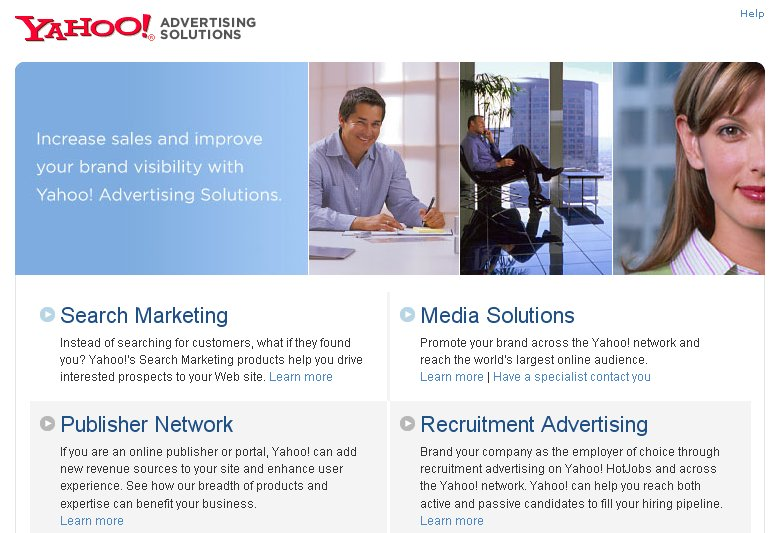
Figure 2 shows a screenshot of Yahoo’s 4P products for the consumer segment for illustration.



Reproduced with permission of Yahoo! Inc. ® 2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo   
are trademarks of Yahoo! Inc.

Figure : Overview of the provider’s products for the consumer segment

There is a second set of products that are almost hidden from the consumers that browse [www.yahoo.com](http://www.yahoo.com). These services are directed at business customers and are shown in .

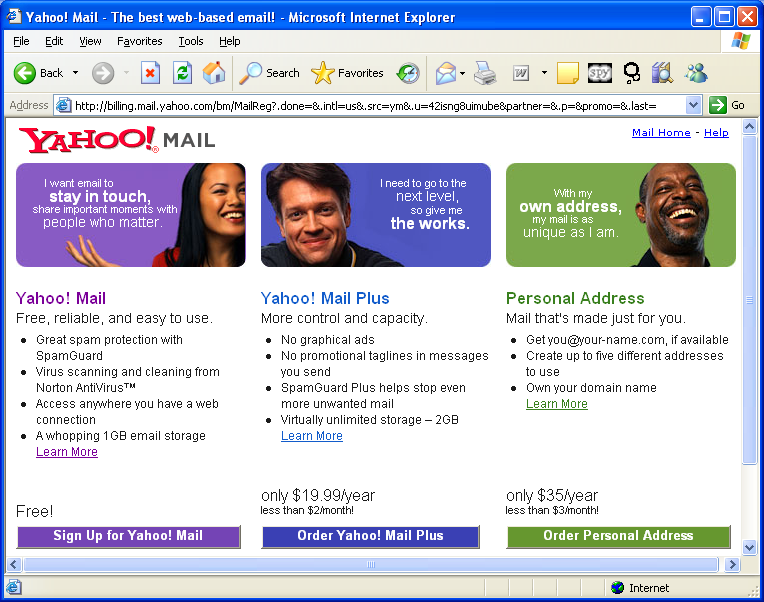


Reproduced with permission of Yahoo! Inc. ® 2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo   
are trademarks of Yahoo! Inc.

Figure : Overview of the provider’s products for the business segment

From the perspective of the 4P Place element of the marketing plan, there are two market segments served by Yahoo, the consumer market and the business market. It could be seen in the two previous screenshots that these two market segments have two completely different lists of products.

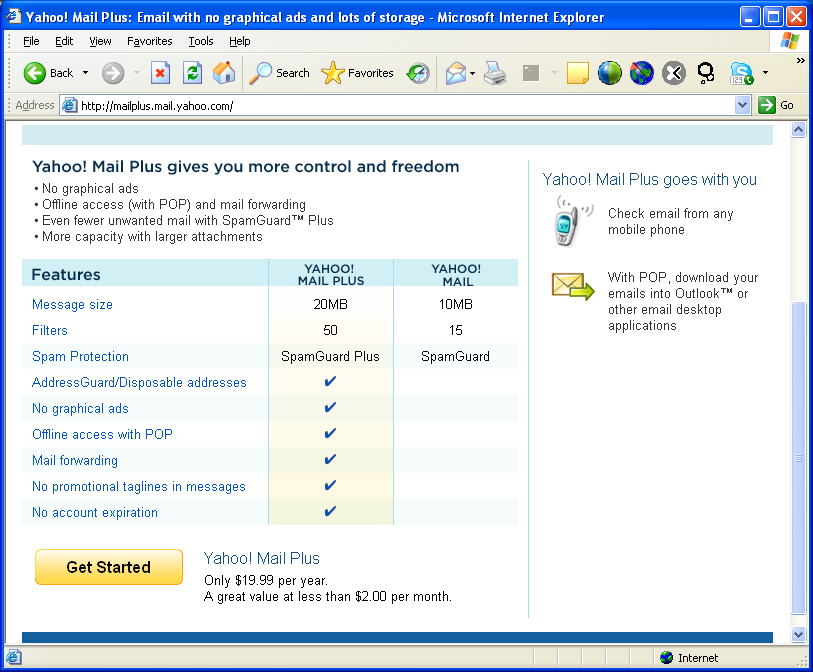
The mail example is based on a case study of Yahoo’s mail products. shows three 4P products and their corresponding 4P pricing.



Reproduced with permission of Yahoo! Inc. ® 2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo   
are trademarks of Yahoo! Inc.

Figure : The provider’s mail products and related pricing

Looking more into detail of the Yahoo! Mail Plus and the Yahoo! Mail products, we find a list of properties that characterize the products in . The identified properties reflect the key value propositions of the product and the differentiation of the product into variants. These properties facilitate market communication by allowing the customer to quickly grasp the differences between the variants, they guide the customer’s buying decision by highlighting the key benefits and allow an up-selling from a basic product towards a more sophisticated product.



Reproduced with permission of Yahoo! Inc. ® 2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo   
are trademarks of Yahoo! Inc.

Figure : Variants of mail are differentiated by properties

The last element of the marketing plan is 4P Promotion, which runs marketing campaigns to promote the provider’s products. Promotion is out of scope for the case study.

### SID Model

An overview of relevant SID classes that are related to the marketing plan are shown in the figure 6. The 4P Place is represented by a SID SalesChannel that serves SID MarketSegments. Each SID SalesChannel provides a number of SID ProductOfferings to the corresponding SID MarketSegments. A SID ProductOffering is the combination of a SID ProductSpecification (the 4P Product) with the SID ProductOfferingPrice (the 4P Price) provided over a SID SalesChannel (the 4P Place). A SID ProductCatalog is a list of SID ProductOfferings.

A SID MarketingCampaign represents the 4P Promotion element of the marketing plan; it is launched across the addressed SID MarketSegments and SID SalesChannels and promotes a subset of the provider’s overall SID ProductOfferings that are specified by a related SID ProductCatalogSpecification.

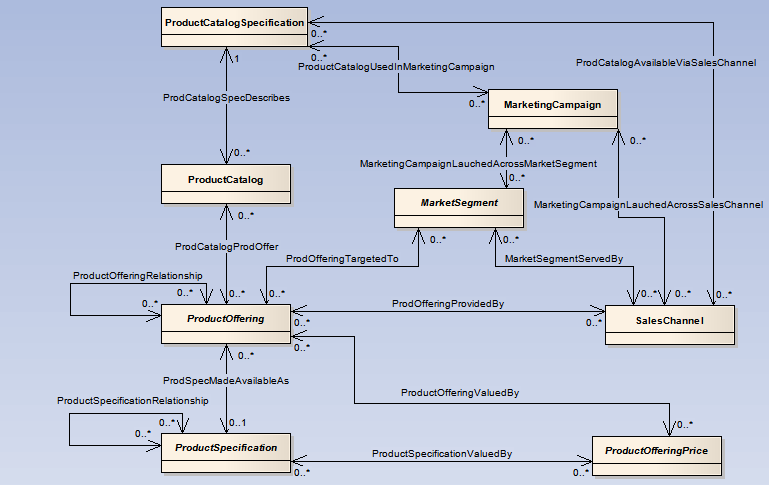


Figure : Selection of relevant SID classes related to the marketing plan

As mentioned earlier, modeling of products/services can be done in different ways, using either the user-consumes-service or the user-consumes-product approach.

#### Product catalog for user-consumes-services approach

An example of the product catalog for the consumer segment, using the user-consumes-service approach is shown in the figure 7. Only the Yahoo! Mail and Mail Plus offerings are shown, the complete catalog of course would contain more offerings than those two.

The SID ProductSpecifications are characterized through SID ProductSpecCharacteristics and SID ProductSpecCharacteristicValues. In the diagram these are not represented by objects, but by a note that is part of the class icon in order to make the diagram easier to read.

The SID ProductSpecifications mail and mailPlus are variants; they have the same SID ProductSpecCharacteristics, but differ in the characteristic values. This means that the model could be improved by deriving the variants from a common template – a technique that has been successfully applied in the MTOSI interfaces.

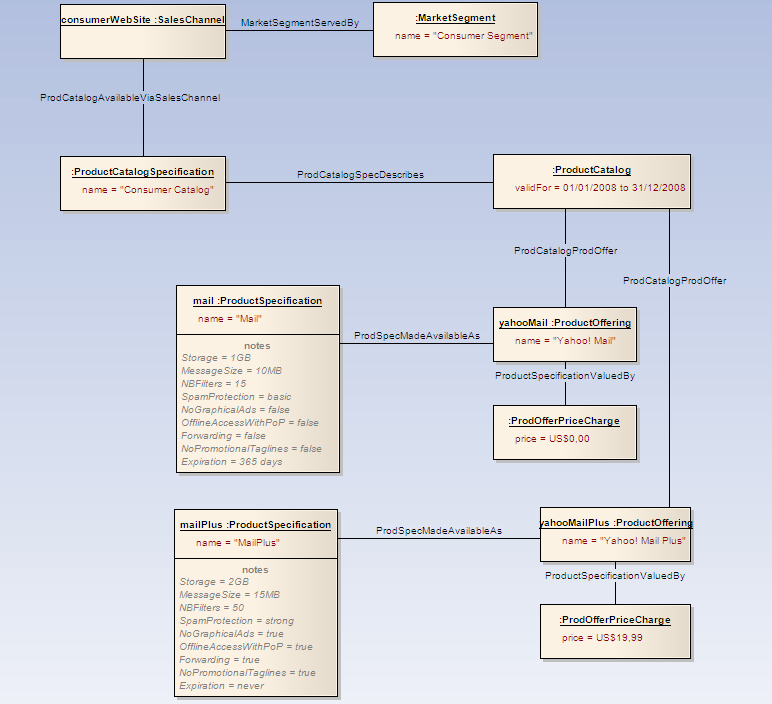
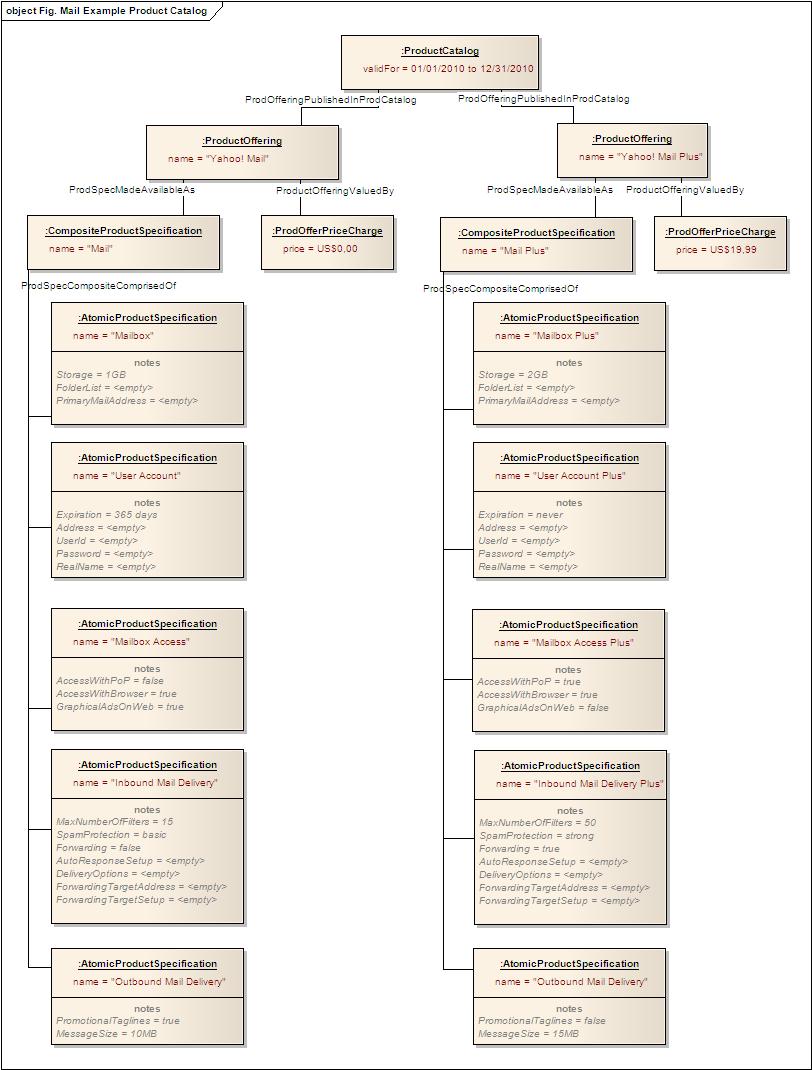


Figure : Example of SID object model for the product catalog for the consumer segment

In the user-consumes-service approach, only those characteristics are present in the product specifications which are relevant for closing the commercial agreement, Characteristics that are related to actually using the mail functionality (e.g. the mailbox address) are modeled on the service layer.

#### Product catalog for user-consumes-product approach

The user-consumes-product approach is presented in two different diagrams. Figure 7a considers that all atomic product specifications (Mailbox, User Account etc) are composite of the product specification, “Mail” and it is mapped to the p roduct offering of ‘Yahoo! Mail’.



< Figure 7a>

In the user-consumes-product approach, product specifications have characteristics representing properties that are relevant for closing the commercial agreement and also properties that are related to actually using the mail functionality.

In the above approach one can observe that it is simple and easy modeling approach but the number of product specifications gets proliferated. The salient points about this approach include.

* Each specification has a single value for each of its characteristics
* In order to show different flavors of a specification, one need to create instances for each possible combination of characteristic values.

The following figure 7b depicts a more elaborate model that optimizes the entities/objects during modeling and implementation by the reuse of characteristics, their enumerations and product specifications.



< Figure 7b>

One can observe that the above approach leads to reuse of Product entities and also using the entities like ConfigurableProductSpecChar.

In this approach the proliferation of product specifications and characteristics can be avoided. Nevertheless UML diagrams’ complexity will grow even more quickly than in the simple modeling of Figure 7b.

When a new ProductSpec is created there is an entry needed to map the respective service spec for order decomposition (mostly done by order management application). Using this approach avoids touching the OSS application as the service layer will not be impacted when new products are introduced using the existing simple products.

However Product Management group will be a major stakeholder to approve model outputs and product catalogue is creations / changes.  With the above approach, it is easier for the product manager to understand how the product offerings are being derived from a reusable definitions (product specifications) decided during modeling time (that is done beneath the level of ProductOfferings).

During the process of documentation the PLM team has found an issue with the above modeling approach. PLM team could not represent/model the control of attribute values being presented to the customer through the product offering at the order time. This is because, at the time of drafting this document refers to SID VIII which does not specify a way to control the ProductSpecCharValues from ProductOffering, This is much needed to control the values of characteristics at the Product Offering level. From the discussion held to draft the views of modeling, different participants agreed that the mapping of ProductSpecCharUsage and ProductSpecCharValueUsage to ProductOffering can resolve this issue. A [change request](http://www.tmforum.org/Community/groups/product_lifecycle_management/changerequests.aspx) is raised on SID VIII

#### Modeling considerations with regard to product offerings

When the product offering becomes complex having some dynamic (configurable) attributes that needs customer’s choice selection, any rules among these selections/options, it is not wise to have a bundle ProductOffering representing all the product definitions. Considering this, another way of modeling is represented in the figures 7c. This modeling approach considers all the atomic ProductSpecifications to have respective Simple ProductOfferings under a bundle ProductOffering of ‘Yahoo Mail’.or ‘Yahoo Mail Plus’.

In this diagram, the ProductSpecCharUse is used to represent the different possible combinations of ProductSpecChar having a relationship association with the respective ProductSpecification. As mentioned earlier the association of these ProductSpecCharUse to the respective ProductOffering helps in controlling the values at the offering level which are to be presented to the customer. Right now in the figure 7c, blue lines represent the expected association of ProductSpecCharUse to the ProductOffering.



<Figure 7c>

The pros and cons of this approach include

Pros:

1. Customer views the constituents (components) of a bundle productOffering
2. Easy selection of the options (dynamic attributes configuration)
3. Most important advantage is during the fulfillment time, the decomposition of the products to services will be simple with respective attributes selections.

Cons:

1. In some cases the navigation and selection leads to a few tabs or scrolling of the pages in respective order handling/capture application (Self-care / CRM / Point of Sales).

### Requirements to Lower Layers

The marketing plan is a tactical management concept for the management of a service provider’s offerings to the market according the marketing strategy. This management viewpoint differs from the operational viewpoint of selling a product to a customer by means of a CRM function.

In order to implement the marketing plan, the service provider needs to create sellable products that actually can be offered, ordered and billed.

On the other hand, it is important to create a marketing plan that makes optimal use of the sellable products that are already in place.

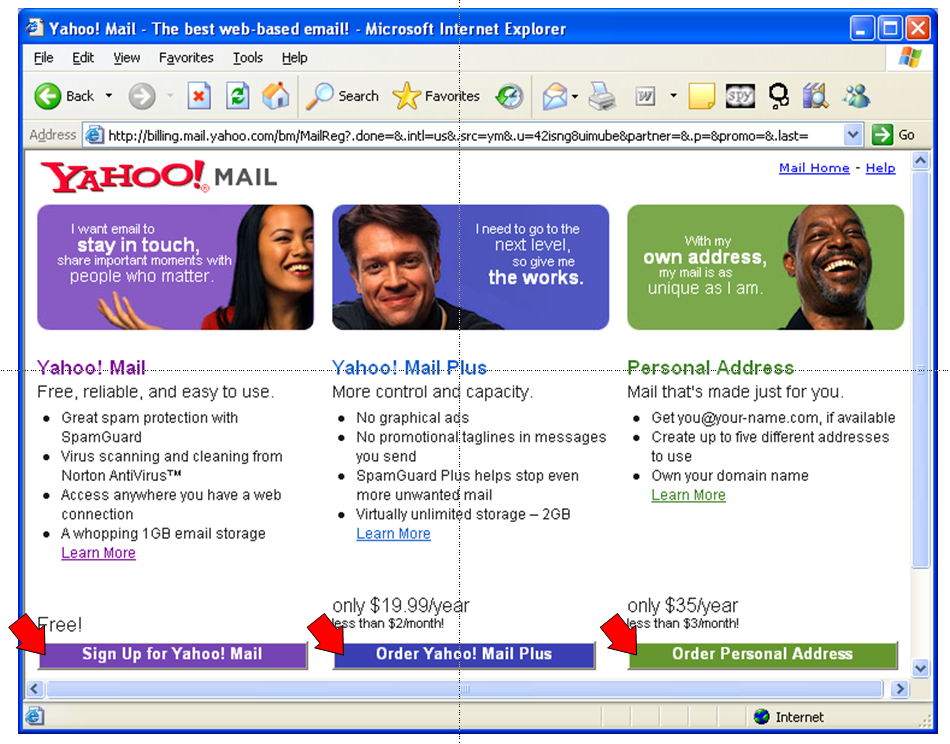
## Sellable Products

### Domain-specific Analysis

The service provider and a customer deal on the level of promises: The service provider promises a service that will be of value to the customer, in turn the customer promises payment of a charge. These mutual promises are the object of a commercial agreement that usually has the form of a written contract underpinned by general terms and conditions.

Nobody can use the promise to send a mail, this only possible by using the underlying services. The eTOM framework is built around the separation of products from services: A product is something which can be the subject of a commercial agreement. In short we could say “customers buy products, users use services”[[2]](#footnote-2). A customer buys (or subscribes to) a product in order to gain access to the value that is provided by the services that are commercially packaged as the product.

During sales, a customer evaluates the service provider’s offering that is a combination of a value promise with the corresponding charge. This can been seen in (view point 1), where the combination of value promise and charge are guiding the customer’s decision to select a button that will lead to an agreement (highlighted with a red arrow).



Reproduced with permission of Yahoo! Inc. ® 2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo   
are trademarks of Yahoo! Inc.

Figure : The customer’s buying decision is guided by value promise and price

Any company must keep track of its current commercial agreements with its customers (in the telecommunications industry sometimes referred to as subscriptions) – the same is true for the service provider. The service provider needs to keep track of the products that its customers have subscribed to.

### SID Model

The agreement between a service provider and an individual customer is captured in a SID agreement. It lists all the products a customer has currently subscribed to. Formally, the SID Agreement consists of a number of SID AgreementItems, each containing a SID Product representing a mutual promise.

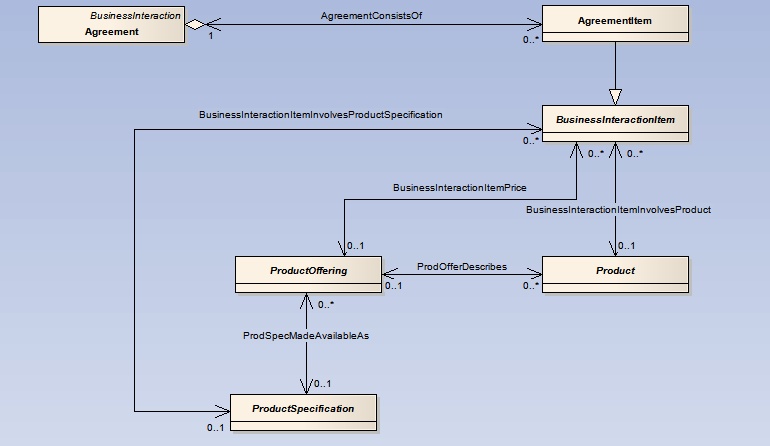


Figure : Selection of relevant classes related to the agreement between provider & customer

In a simple example for the agreements of the service provider with the customers Paul, Michel and Tony is shown. Michel has a subscription for Yahoo! Mail, Paul has a subscription for Yahoo! Mail Plus, and Tony has a subscription for Yahoo! Mail Plus and Yahoo! Personal Address.

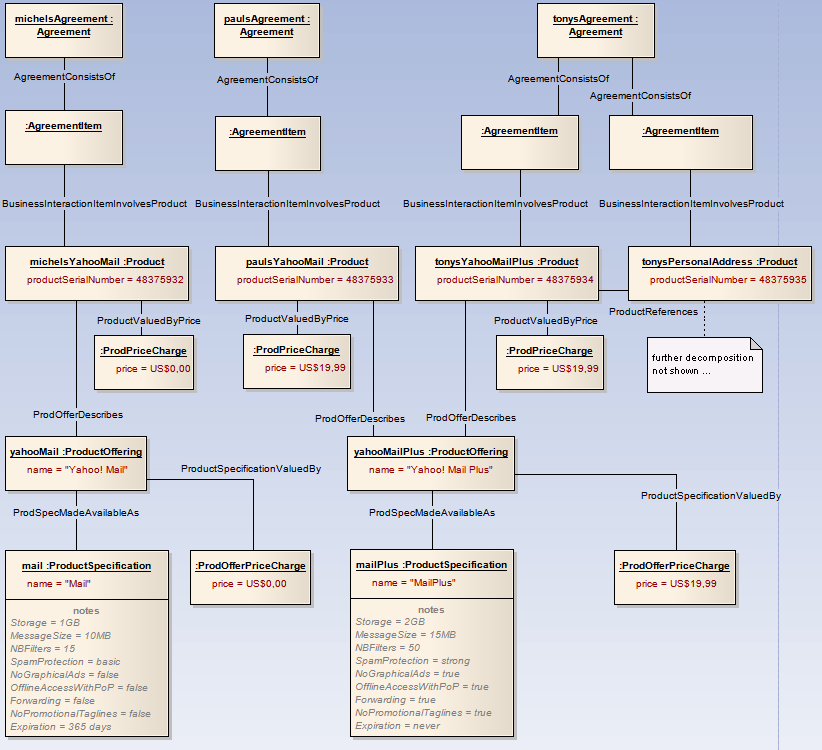


Figure : Simple example of SID object model for the agreements of the provider with three customers

The above figure (10) shows the product instantiation (SID Product) for the user-consumes-service approach. The instantiation of individual product instances reflects the commercial aspects of the agreement between customer and provider.

The following two figures (10a, 10b) show the product diagrams for the user-consumes-product approach.

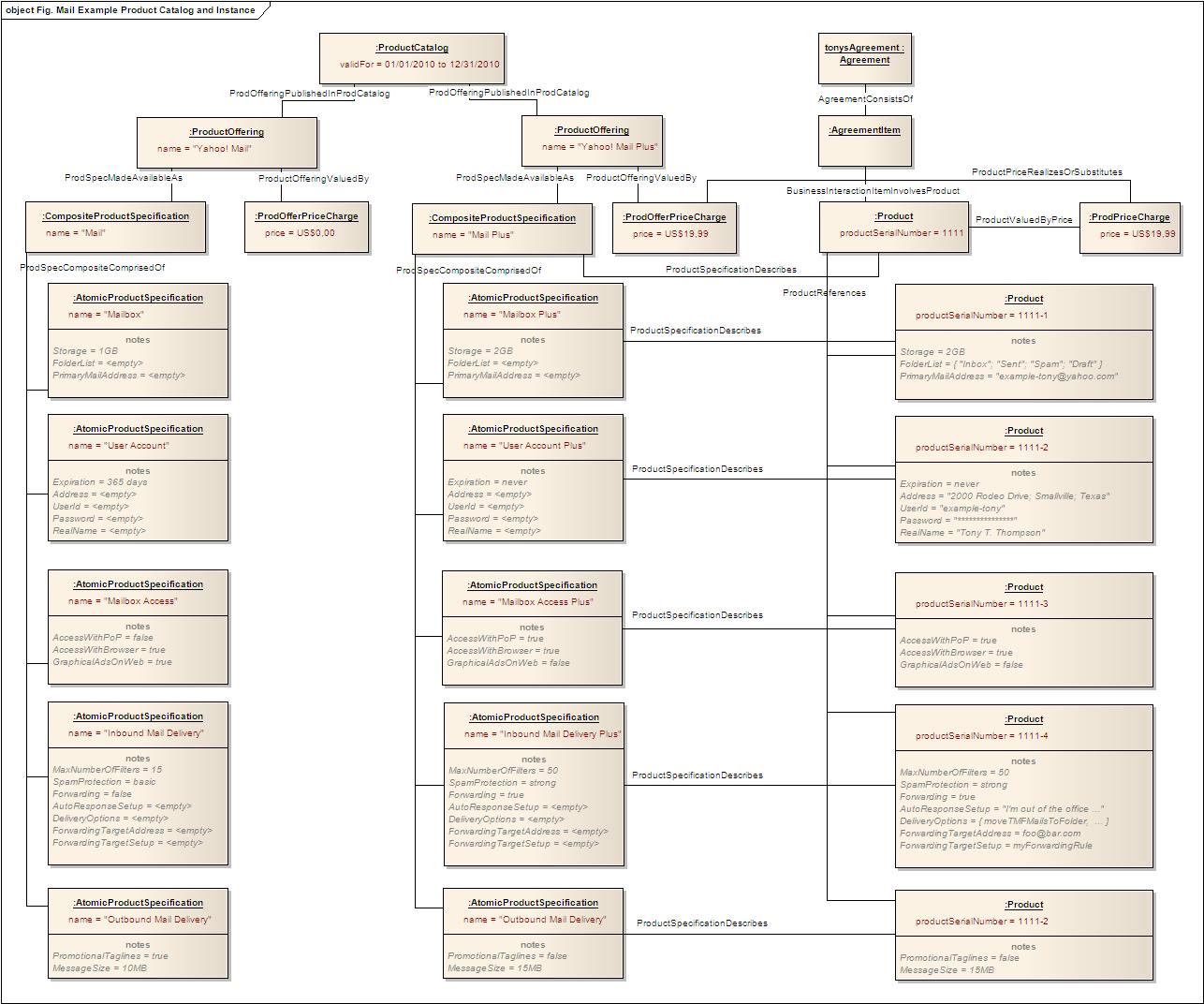


Figure 10a

Figure 10a shows how Tony’s product instances capture usage-specific details in the characteristics (e.g. the primary mail address [example-tony@yahoo.com](mailto:example-tony@yahoo.com), which is not the case in the user-uses-service approach.



Figure 10b

### Requirements to Lower Layers

The product specifications define what is promised to the customer as part of the agreement. Therefore the product specifications are the requirements for the underlying customer-facing services that need to be provided by the service provider in order to fulfill what is promised at the product level.

## Customer-facing Services

### Domain-specific Analysis

For the purpose of identifying the customer-facing services, the service provider is regarded as a system with a concise system boundary. It is the purpose of that system to provide value to stakeholders that are considered “external” to the system. Services represent logical functionality of the system that is exposed to stakeholders through interaction points.

Use cases are a well-established tool to analyze the behavior of a system that is exposed to stakeholders. In this section, we will perform a use case analysis in order to identify the services of the system “Extended Service Provider” from the viewpoint of the external organizations and individuals. We called it “Extended Service Provider” instead of “Service Provider” because

* the northbound system boundary runs through customer-facing applications that are hosted on customer equipment, i.e. the user’s mobile handset running a web browser
* we also consider the role of third party provider at the southbound system boundary

These services represent the customer-facing services provided by the service provider: What value does the provider deliver to organizations or individuals that they actually pay for? What does a user know about the particular interaction points to gain access to that value?

The analysis of customer-facing services helps to identify what really matters to organizations or individuals, focusing on the perceived value and not on the technical implementation. A user doesn’t care about the underlying technical protocols for communication between his interaction point and the provider’s infrastructure. The protocols could even be replaced as long as the user’s perspective remains the same: The set of customer-facing services that sum up the customer experience of mail can be realized through different underlying transport and presentation protocols.

The following use case diagram shows the key use cases of the system “Extended Service Provider” for our mail example. They represent the functionality that needs to be provided in the form of customer-facing services in order to realize the value promise given at the level of sellable products.

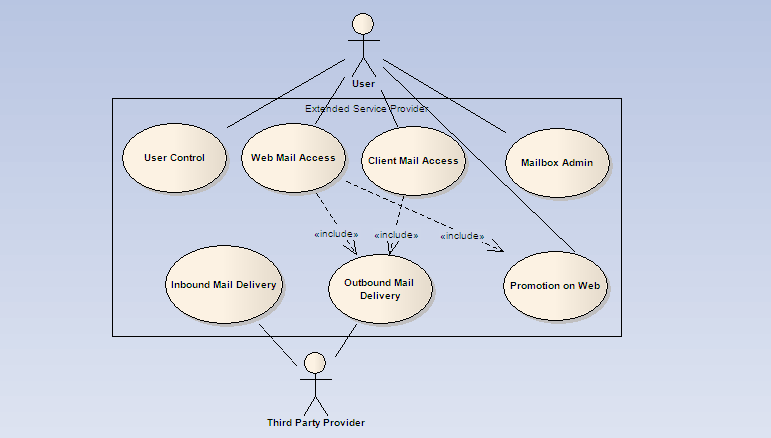


Figure : Key use cases of the extended service provider

Strictly speaking, only the northbound services are customer-facing, because the Third Level Provider is not a primary customer role, but rather a supplier / partner role.

The use cases are as follows[[3]](#footnote-3):

* User Control. Validate username and password for subsequent web access, handle the blocking and expiration of user accounts
* Web Mail Access. Create and send mails to recipients in the internet and read mails from the mailbox by means of a standard web browser connected to the internet
* Promotion on Web. Display promotions that match keywords in current mails
* Client Mail Access. Create and send mails to recipients in the internet and read mails from the mailbox by means of a standard mail client application connected to the internet
* Mailbox Admin. Manage the profile of the mailbox that controls the mailbox functionality for sending and receiving mail by a standard web browser connected to the internet, e.g. filters, signatures, auto-response
* Inbound Mail Delivery. Receive and store a mail in a local mailbox that is transmitted by a third party provider on behalf of a mail sender
* Outbound Mail Delivery. Transmit outbound mail to a third party provider

User Support (Help Desk Services) is out of scope for this example, it is a candidate for another customer-facing service.

The use cases are detailed into a description of the customer-facing services that shall be packaged and sold as a product.

|  |  |
| --- | --- |
| Customer-facing Service | Interfaces and Operations |
| User Control | User Login   * Validate username and password for subsequent web access * Block access after <n> unsuccessful login attempts * Block access, if account has expired   User Administration   * Unblock access for forgotten password and blocked/expired access by means of challenge with secret question * Change password * Manage profile of real name, postal address, interests |
| Web Mail Access | Send Mail   * Create and format mail * Add signature from mailbox profile * send mails to recipients in the internet * Maximum message size <n> * Add promotional tagline to outgoing mails <true/false>   Read Mail   * Read mails, move to folder and delete |
| Promotion on Web | Promotion   * Display promotion according to keywords in current mails   + Promotion <on/off> |
| Client Mail Access | Send Mail   * Create and format mail   + Add signature from mailbox profile * send mails to recipients in the internet   + Maximum message size <n> * Add promotional tagline to outgoing mails <true/false>   Read Mail   * Read mails, move to folder and delete |
| Mailbox Admin | Mailbox Administration   * Set signature to be automatically added to each outgoing mail * Set text of vacation-response to incoming mail, set auto-response on/off * … |
| Inbound Mail Delivery | Mail Transmission   * Deliver incoming mail to standard inbox folder * Deliver incoming mail to existing folder according to filter setting * Auto-respond to incoming mail according to auto-response setting |
| Outbound Mail Delivery | Mail Transmission   * Forward outgoing mail to internet |

At the first glance, mailbox seems to be an obvious candidate for a customer-facing service, because this is the fundamental concept that the user is aware of. The use case analysis showed that mailbox is not a use case in its own right, but a business entity used in the description of different use cases. Therefore mailbox has not been identified here as a customer-facing service.

#### Modeling Systems and Services with UML

In the next sections, we apply some established practices of systems engineering in order to model the technical functionality of the mail example. The concept “system” helps us to define the functional requirements of the system to be built and to detail these requirements step by step into an implementation at the resource level.

A system[[4]](#footnote-4) is a collection of building blocks that collaborate in order to achieve a goal that can’t be achieved by one of the single building blocks alone. A building block may consist of software, hardware, persons or other entities.

A building block (in short “block”) has a boundary that encapsulates and hides its internal construction to its environment. Blocks are modular units of system description. Each block defines a collection of features to describe a system or other element of interest. These may include both structural and behavioral features, such as properties and operations, to represent the state of the block and behavior that the block may exhibit. In order to provide access to the realized behavior, the block defines a number of ports as interaction points.

A port[[5]](#footnote-5) specifies the services that the owning block provides (offers) to its environment as well as the services that the owning block expects (requires) of its environment. The specification of the services is achieved by typing the standard port by the provided and/or required interfaces. An interface defines a set of related operations that a stakeholder can invoke to use the service. In the communications industry, a port is often referred to as service access point.

*Note: According to OASIS, a service is a mechanism to enable access to a set of one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description. The consequence of invoking a service (interaction) is a realization of one or more real world effects. This definition is compatible with the UML/SysML style definition, but it avoids any reference to the block.*

UML component diagrams can be used to depict blocks, ports and interfaces ().

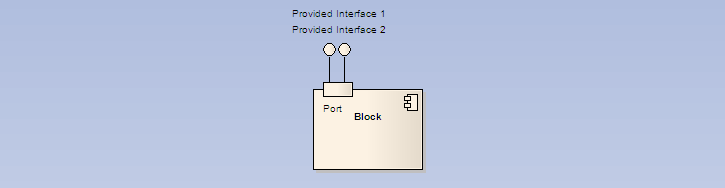


Figure : UML component diagram depicting a block, a port and interfaces

The service that the block provides at a port (a service access point) is specified by typing the port by interfaces. A service is linked to an individual port because the port has the purpose of providing a service to the environment, but it is not depicted as an object in its own right in the diagram. Because we are using services as key entities in lifecycle management, we need a way to depict the services as individual entities in our models.

Therefore the following modeling style is applied in the following chapters (, left): A service is modeled as an UML component of stereotype <<service>> that is attached to a port by means of a dependency of stereotype <<exposes>>. An external stakeholder uses a service that is provided by the block by invoking operations at a port (the service access point) that conform to the prescribed interface definition. On the technical level, the invocation of operations is often exercised by means of sending messages to the port and by receiving corresponding result messages.

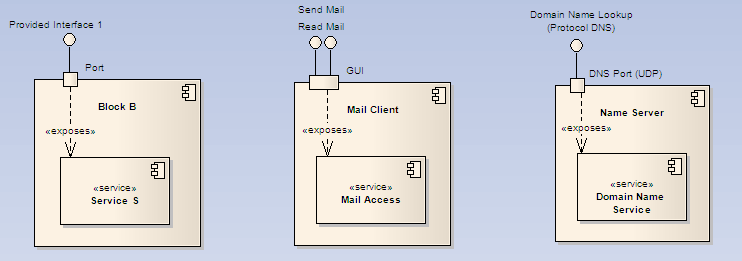


Figure : Using the UML component diagram to show how services are exposed at a port

A mail client application provides a mail access service to a human user (, middle): The human user interacts with the mail access service by executing operations to send, retrieve, move and delete mails at a graphical user interface (GUI). The mail access service is modeled as a UML component of stereotype <<service>>, the graphical user interface corresponds to a UML port and the functions provided to the user at the GUI correspond to the UML interface.

A DNS name server usually provides the domain name service at UDP port 53 (, right). A service requestor interacts with the DNS service by sending messages to that port and by receiving messages from that port. The messages must conform to the DNS protocol which defines a set of DNS queries that can be exercised. The domain name service is modeled as a UML component of stereotype <<service>>, the UDP port corresponds to a UML port, and the set of DNS queries defined in the DNS protocol corresponds to the UML provided interface.

The UML component diagram in is a translation of the use cases identified before. It depicts the customer-facing services as UML components, the interaction points as UML ports and the sets of operations made available externally as UML interfaces. The interaction involves an information flow between the external actors and the interaction points of the Extended Service Provider.

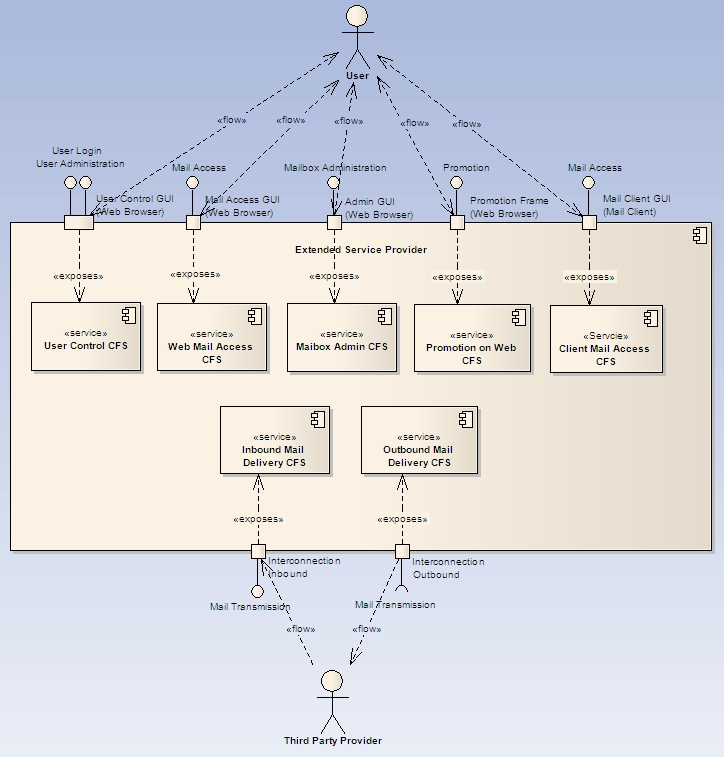


Figure : Translation of the use cases to a UML component diagram of customer-facing services

#### Business Entities that complement Customer-facing Services

In the preceding chapter, customer-facing services have been identified by means of use cases, thus focusing on the behavioral aspects of the system “Extended Service Provider”. In systems analysis, use cases are complemented by the business entities that the use cases act upon. We will follow this approach and identify the business entities that are the basis of the behavior exposed through customer-facing services.

The business entities are the concepts that a user knows about when using the customer-facing services, e.g. a user will say “I’m going to check my mailbox”, thus referring to the business entity mailbox. shows the business entities that are the objects that the customer-facing services operate upon. The customer-facing services are a façade towards the user that expose details of the business entities.

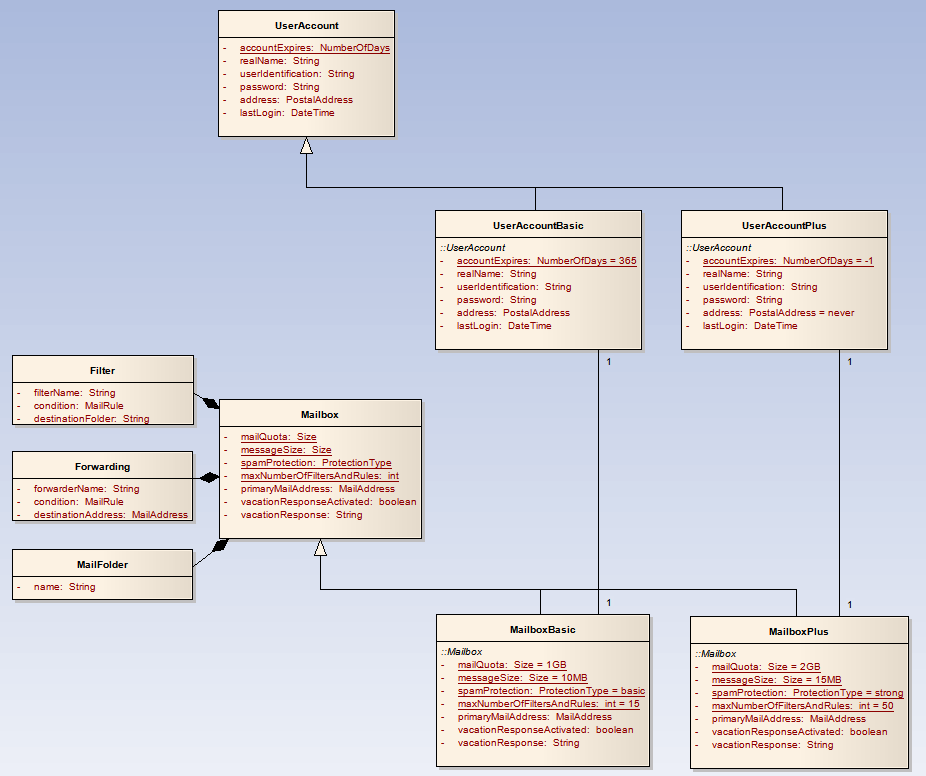


Figure : UML class diagram of the business entities in the context of the products Mail and MailPlus

A UserAccount represents the personal account that the provider creates for a user. The two variants UserAccountBasic and UserAccountPlus differ in the expiration time, UserAccountBasic expires after 365 days of inactivity, while a UserAccountPlus never expires.

A Mailbox represents the mailbox in which incoming mail is stored to be picked up by a user. The two variants of Mailbox are MailboxBasic and MailboxPlus. There are two different kinds of access to that mailbox, the Web Mail Access CFS and the Client Mail Access CFS.

### SID Model

In the next step, a SID model for the customer-facing services is developed.

The SID model in consists of two sections that contain functional and wrapper customer-facing services:

* functional customer-facing services are the services that have been derived by the use case analysis in , they represent a well-defined functionality that can be accessed at the system boundary
* wrapper customer-facing services are wrappers for the business entities that have been identified in

Two SID ServicePackagesSpecs have been created, a cfss\_Mail\_Package and a cfss\_MailPlus\_Package. Each aggregates the set of SID CustomerFacingServiceSpecs that is needed for the provisioning of one of the variants Mail and MailPlus.

Wrapper customer-facing services have been introduced in order to maintain the desired step-by-step decomposition from customer-facing to resource-facing services to logical and physical resources without creating a short-cut directly from products to logical resources.

In the first modeling versions, the business entities Mailbox and UserAccount had been directly mapped to logical resources. In that case, there was no need for wrapper services, but the strict hierarchy of step-by-step decomposition had been broken. This modeling alternative had been considered problematic from a viewpoint of the MTOSI interface design which mandates a strict hierarchy product – service – resource, and had therefore been discarded.

From the viewpoint of how the mail technology works the concept of wrapper services is a somewhat artificial and initially confusing workaround. A strict hierarchy is a natural modeling choice for classical network-oriented technology. There the provisioning of a product requires the programming of network devices that as a consequence expose new services that are dedicated to a customer and that had not been in place before.

This is different for content-oriented services like mail. There the only thing that really is provisioned are the logical resources, services are shared and need not to be programmed to support a particular product. Section discusses the sharing of resource-facing services between multiple clients in detail. In addition, there are multiple access services (e.g. Web Mail Access, Client Mail Access) that provide access to the same logical resource (e.g. Mailbox), so definitely the Mailbox has an identity of its own. Therefore it seems not to be a good modeling choice to incorporate the Mailbox into one of the access services.

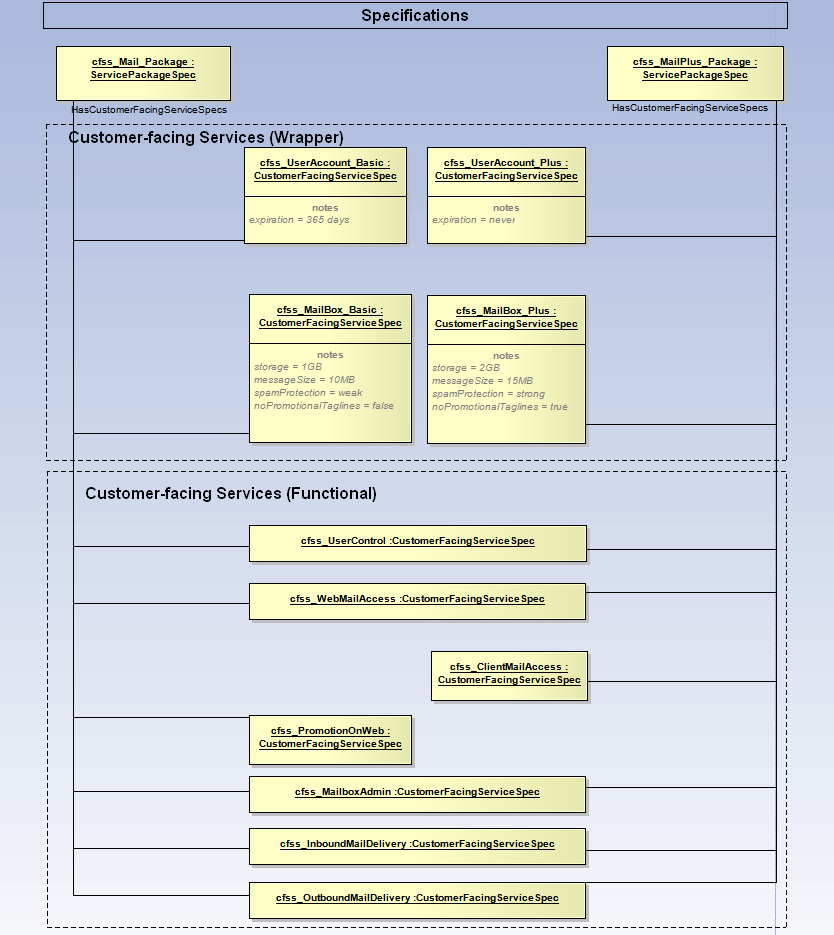


Figure : SID model of the specifications for customer-facing services for the mail example

shows how entities are related to specifications at the level of customer-facing services for the example of Tony subscribing to MailPlus.

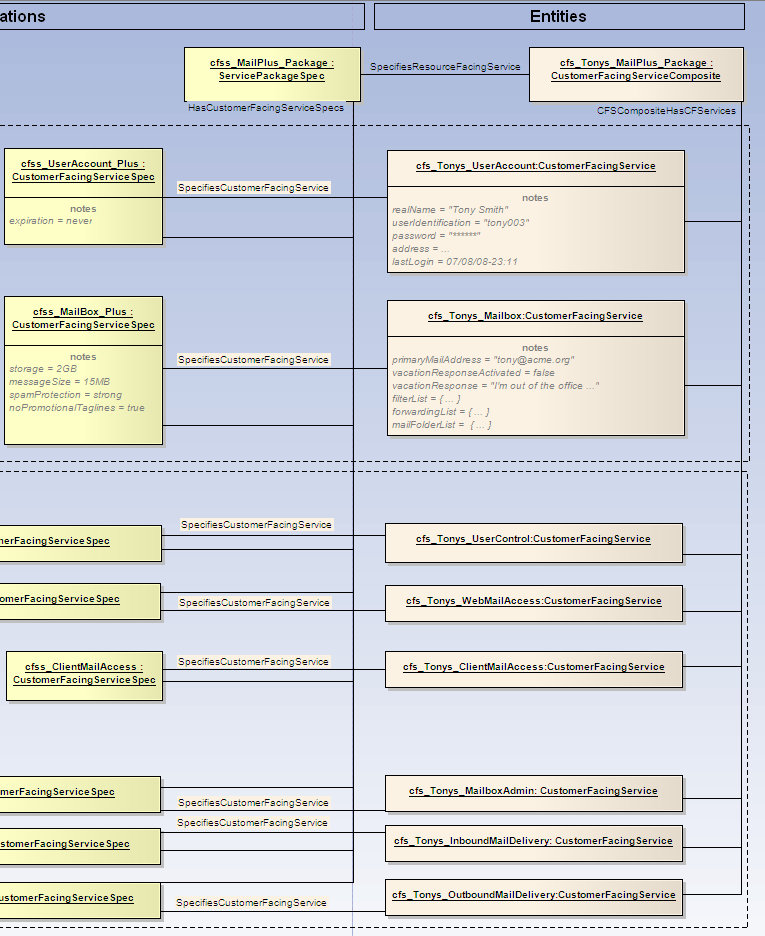


Figure : SID object model of the entities and specifications for customer-facing services for the mail example

### Requirements to Lower Layers

The customer-facing services define what a user will see without looking into the technical protocol implementation between terminal equipment and provider infrastructure.

Terminal equipment accesses the functionality that is implemented in the provider infrastructure by means of resource-facing services, therefore the customer-facing services are the source of requirements for the use of resource-facing services.

## Linking Products with Customer-facing Services

### Domain-specific Analysis

In the two preceding chapters we identified the functional and wrapper customer-facing services for the mail example. Now we will describe the assembly of products from these customer-facing services.

### SID Model

The following SID class diagram shows the specification classes for the assembly of SID ProductSpecifications from SID CustomerFacingServiceSpecs and SID ResourceSpecifications.

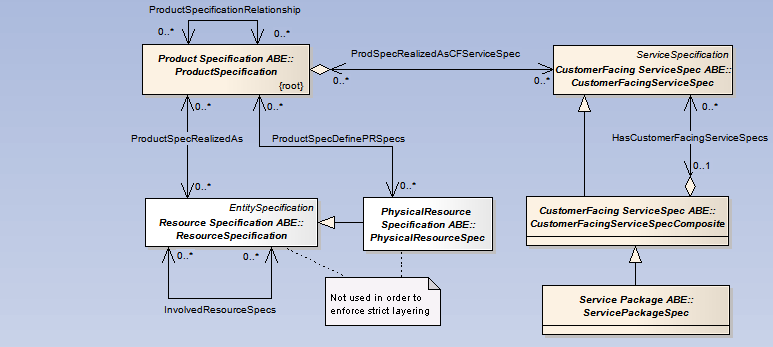


Figure : SID class diagram showing the relationship between selected classes that are related to specifications

shows that the SID allows the construction of products both from customer-facing services and from resources. We will create products on the basis of customer-facing services only. The possibility to directly map from products to resources is intentionally not used. The reason is that we want to enforce a strict layering that avoids shortcuts between layers that are not neighbors in order to allow a step-by-step decomposition in a style that facitilitates the use of MTOSI interface specifications.

The following object diagram shows the content of the catalog that has been identified up to now. Again a schematic modeling style is applied, which represents the SID Characteristic and CharacteristicValue classes as notes.

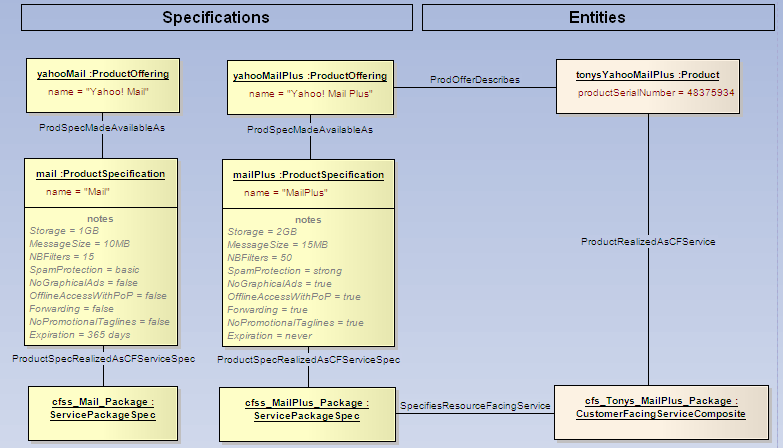


Figure : SID object model of product and cfs specifications related to the mail example

shows that there are two product specifications that have different characteristics on the CRM level. These are the characteristics that the provider exposes to the market, so a customer can select between the two variants during his buying decision.

By ordering a product, the customer gains access to a set of related customer-facing services that are packaged into a common service package. The services contained in the service package expose the mail functionality to the customer through a number of operations that he can execute at specific interaction points.

Figure 19a shows the mapping of products and CFS Specifications for the user-consumes-product approach. In this approach since all the usage-specific characteristics are present on the product layer, the instance-specific characteristic values are propagated from the product layer to the service layer. This usually involves a translation of the characteristic names and of the enumeration values. For the sake of simplicity this is not shown in the example.

This is a bit awkward from a data modeling point of view, because we have a duplication of the characteristic values on the product layer and the service layer, which is a very denormalized model.

From an implementation perspective this translation is a choice, depending on whether application boundaries dictate this. However, characteristics can be shared by employing a generalized characteristic model, such as that in the Common Business Entities domain, rather than having a separate set of implemented tables, one set for Products and one set for Services.

Nevertheless this is the situation in many real-world systems, where the selling and the configuration is done in terms of products on the BSS layer, while the actual production is handled by OSS systems that expose customer-facing services to the BSS layer. In this case the information is actually duplicated in both the CRM inventory and the OSS service inventory, with service orders or service order requests propagating the characteristic values downstream from BSS to OSS.

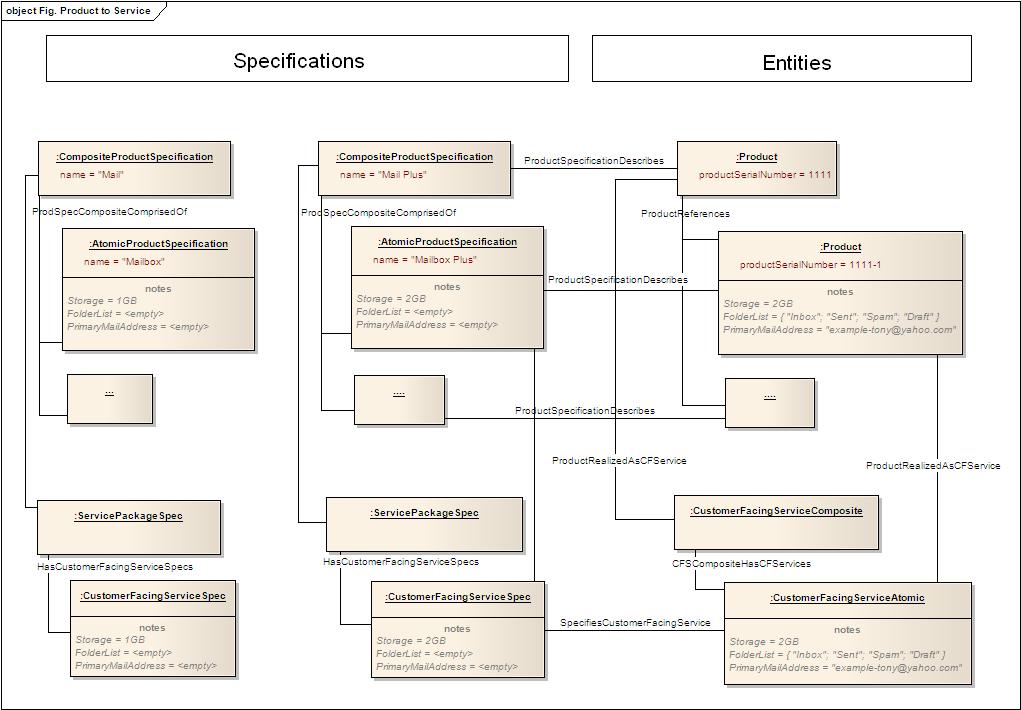


Figure 19a

Figure 19b shows the mapping of products and CFS Specs for the second view point [approach 2.2](#_Approach_2.2:), discussed in the section (2.1.2) and figures 7b and 7c.



< Figure 19b>

## Resource-facing Services

### Domain-specific Analysis

The preceding analysis of customer-facing services provides an excellent requirement base to design the technical interfaces of the infrastructure. While the customer-facing services focus on the organizations and individuals as actors, with the customer-facing applications hosted by the terminal equipment as interaction point, now we move one level deeper into the communication between the customer-facing application and the provider infrastructure.

In this section, we will identify the technical services that the service provider infrastructure provides to the external terminal equipment.

These services represent the resource-facing services provided by the service provider’s technical infrastructure: What service access points does the infrastructure of the service provider provide to the external terminal equipment to connect to?

The analysis of resource-facing services helps to identify all technical ports that need to be designed in order to support a customer-facing service. The resource-facing services with the corresponding technical ports provide an important point of decoupling between the terminal equipment and the provider infrastructure: As long as the resource-facing services remain constant, the hardware and software implementation of terminal or provider infrastructure can be replaced without impacts on the user functionality.

As a starting point, each customer-facing service is analyzed on the technical layer to identify the underlying technical interaction points and the protocols. This is demonstrated for the different customer-facing services in .

A closer look at the User Control CFS finds that the customer experience is created at the browser GUI as interaction point. The User Control CFS is technically realized through an underlying resource-facing service in the service provider infrastructure, the User Control RFS. The infrastructure exposes the functionality of the User Control RFS at the user control port that can be accessed through specific URLs, e.g. http://<ip-address>/login and http://<ip-address>/login/admin. The browser invokes operations on that port by exchanging messages according to the HTTP protocol with the infrastructure.

On the left-hand side of , a 1:1 mapping between customer-facing services and resource-facing services is shown. The 1:1 mapping is just a special case, in many cases a customer-facing service is realized by more than one resource-facing service. An example of a 1:N mapping can be observed on the right-hand side of , where the Client Mail Acess CFS is realized by three resource-facing services.

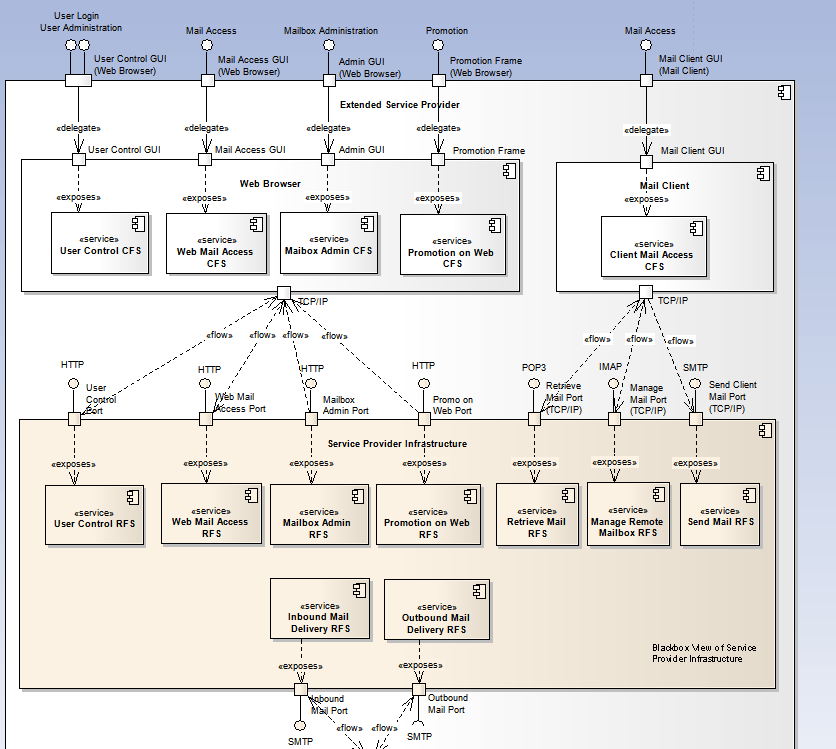


Figure : UML component diagram: Relationship between service provider infrastructure and customer equipment

The result of the analysis is a black box view of the service provider infrastructure that lists the resource-facing services and the related ports and protocols. The following list details the functionality that needs to be implemented by the technical infrastructure of the provider.

|  |  |  |
| --- | --- | --- |
| Resource-facing Service | Description | Protocol |
| User Control RFS | User Login. The infrastructure provides a service to post a username/password HTML page to receive and validate the returned username/password combination and to establish a cookie-based user session  User Administration. The infrastructure provides a service to post an HTML page for the unblocking blocked or expired user accounts | HTTP |
| Web Mail Access RFS | The infrastructure provides a service to post a set of related HTML pages for sending and retrieving mails, requiring an established user session | HTTP |
| Mailbox Admin RFS | Mailbox Administration. The infrastructure provides a service to post a set of related HTML pages for setting the configurable properties of a mailbox, requiring an established user session | HTTP |
| Promotion on Web RFS | The infrastructure provides a service to post context-based advertisements as HTML frame to be embedded into an overarching HTML page | HTTP |
| Manage Remote Mailbox RFS | The infrastructure provides a service to read, move and delete inbound mails which are stored in folders of the mailbox in the service provider’s infrastructure | IMAP |
| Retrieve Mail RFS | The infrastructure provides a service to list, retrieve and delete inbound mails from the mailbox in the service provider’s infrastructure | POP3 |
| Send Mail RFS | The infrastructure provides a service to submit outbound mails for transmission to recipients identified by mail addresses | SMTP |
| Inbound Mail Delivery RFS | The infrastructure provides a service to third party providers infrastructures to submit mails with addresses hosted by the provider | SMTP |
| Outbound Mail Delivery RFS | The infrastructure requests a service to third party provider infrastructures to forward mails within the internet | SMTP |

Note that for the sake of simplicity, authentication for client mail, secure HTTP and DNS are not considered in this example.

### SID Model

Resource-facing services are modeled by the SID ResourceFacingServiceSpec and SID ResourceFacingService classes. This is shown in the following object diagram.

Like in the model for the customer-facing services, we differentiate between functional and wrapper resource-facing services:

* The functional resource-facing services do have a port over which they can be accessed by means of a protocol
* The wrapper resource-facing services are just a wrapper for business entities to be passed through

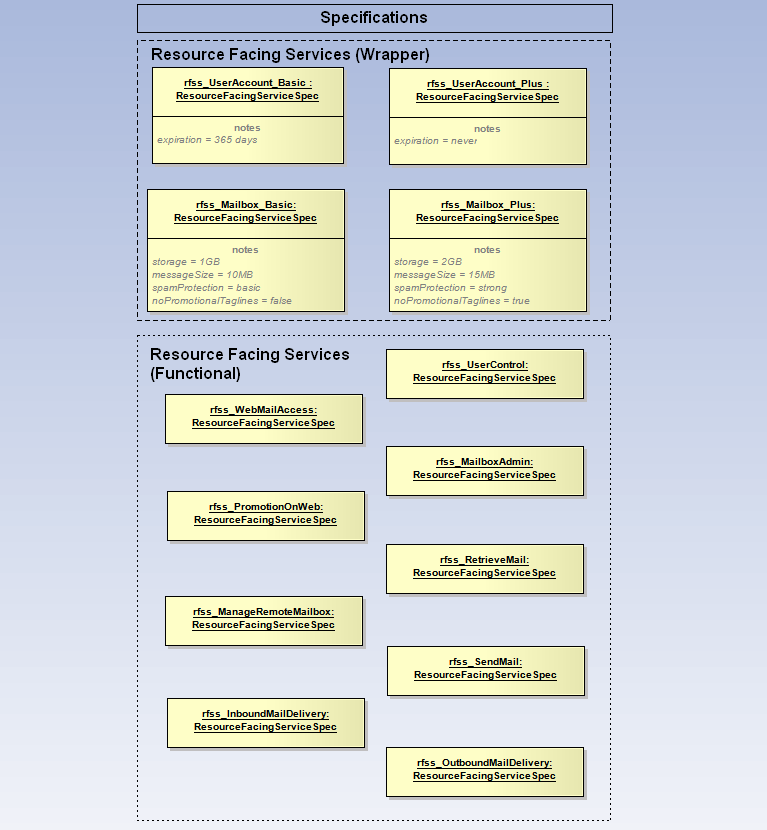


Figure : SID object model of resource-facing services for the mail example

### Requirements to Lower Layers

The resource-facing services define the protocols and functionalities that need to be implemented by a set of logical resources, i.e. software processes.

## Linking Customer- and Resource-facing Services

### Domain-specific Analysis

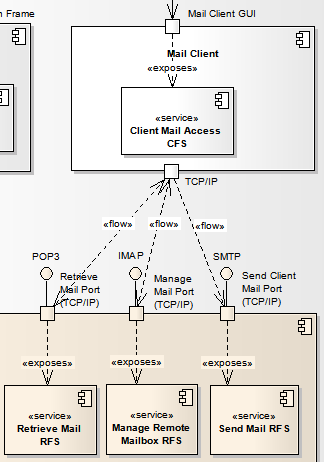


Figure : Ports can be shared between many mail clients

In order to link customer-facing services and resource-facing services, we need to think about which services are individual to a user and which are shared by many users.

In the example of mail, there is not an individual port for each mail client, but a port is shared and can serve many mail clients. A mail client would establish a TCP session with a port exposing the Manage Remote Mailbox RFS and communicate by means of the IMAP protocol.

Therefore in a simple model we have multiple Client Mail Access CFS instances (one per user) that share a singular Manage Remote Mailbox RFS.

It is important to understand that with technologies where many clients share the singular resource-facing services of a platform there is always such an N:1 relationship (or N:2 in case of simple redundancy). Maintaining the link enables us to derive the customer-facing services per customer that are affected from a failure of a singular resource-facing service, thus improving the capability of the help-desk to identify the root cause of trouble reported by a customer.

### SID Model

In this section we will apply SID modeling to show how customer-facing services are built on top of resource-facing services.

shows the mapping of customer-facing services to resource-facing services for the two wrapper customer-facing services user account and mailbox. The mapping is a 1:1 pass-through mapping. For each user account and mail box entity at the CFS level there is a corresponding entity at the RFS level – the entities are dedicated and not shared among different users.

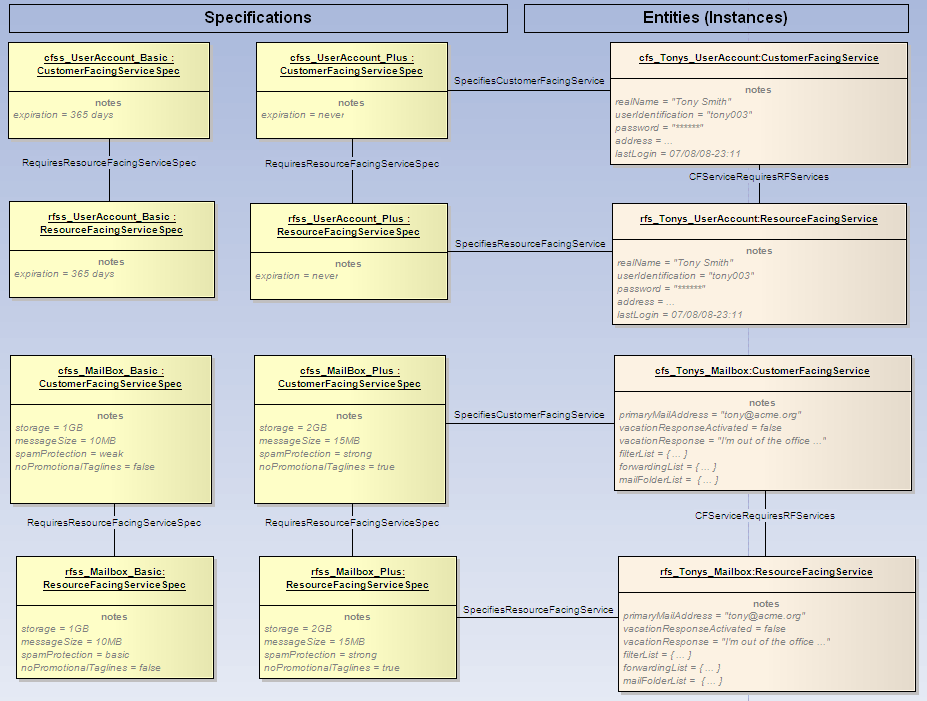


Figure : SID object model of the mapping of customer-facing services to resource-facing services for UserAccount and Mailbox

shows the situation where a single customer-facing service decomposes into a number of resource-facing services by the example of cfss\_ClientMailAccess. The three resource-facing services on the entity side are shared among multiple instances on the customer layer: Both Paul’s and Tony’s client mail access customer-facing service map to common shared resource-facing services.

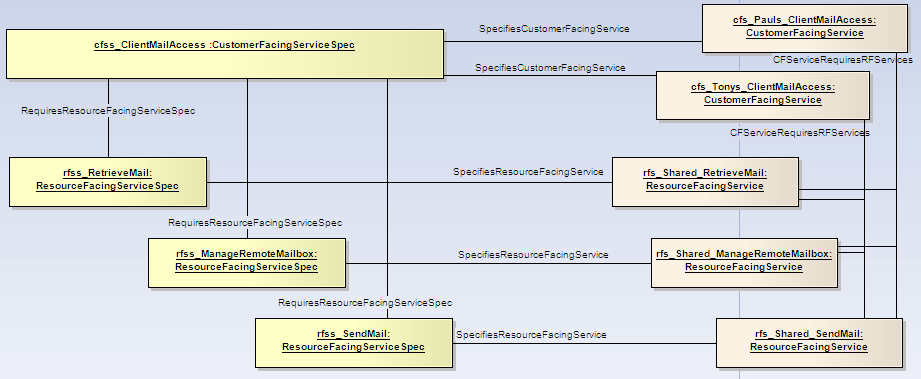


Figure : SID object model of the mapping of customer-facing services to resource-facing services for ClientMailAccess

## Logical Resources of the Infrastructure

### Domain-specific Analysis

In this section, we will construct a specific infrastructure that implements the system described by the black-box view of resource-facing services, ports and protocols.

A public domain implementation using the packages Postfix, Dovecot, Squirrel, Apache and Mysql was chosen, it is depicted in . Of course this is not the infrastructure underlying Yahoo’s products, this implementation has been chosen for the purpose of illustration because it is a real one that is simple enough to be depicted in a single diagram.

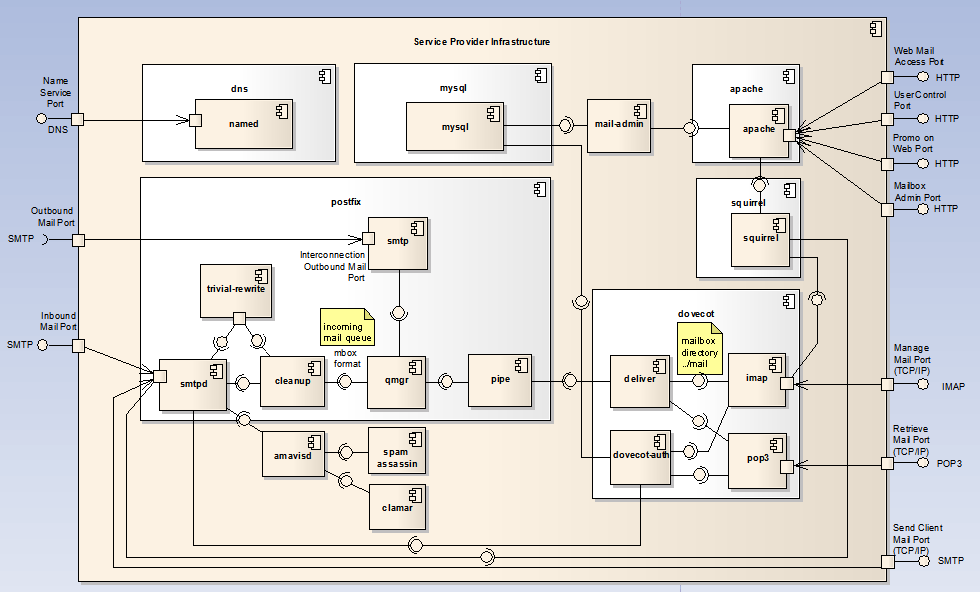


Figure : UML component model of the service provider infrastructure at the logical resource level

shows the white box view of the service provider infrastructure. The infrastructure consists of quite a number of interconnected software processes which are modeled as UML components. The ports of the service provider infrastructure are the same as in the black box model in the previous section (). Each port at the external system boundary is mapped to a port of a corresponding software process by means of the UML <<delegate>> relationship. Please note that the direction of the arrow does not mean the direction of an information flow, but just the delegation of the port implementation to a lower level component.

An inbound mail addressed to the Inbound Mail Port is received by the smtpd process and handed over to amavisd for subsequent spam checking by spam-assassin and virus checking by clamar. Trivial-rewrite adjusts the mail headers, then the mail is cleaned up by cleanup which writes the mail in the mbox format into the incoming mail queue. Qmgr reads the incoming mail queue, interprets the mail recipient addresses and decides whether to forward the mail via smtp to external service providers through the Outbound Mail Port. In case of a local recipient, qmgr hands over the mail to the deliver process by means of the pipe process. Deliver writes the mail into the mailbox of the recipient, which is a directory (e.g. …/mail) of files where each file represents a single mail.

A mail client retrieves mails from the mailbox by connecting to the Retrieve Mail Port and interacting with the port by means of the POP3 protocol. In response to the POP3 operations invoked by the mail client application, the pop3 process reads the mails stored in the mailbox directory, sends them to client and deletes them from the mailbox directory.

The imap process is an implementation of the IMAP protocol; it allows managing mails that remain in the mailbox. The difference between IMAP and POP3 is that POP3 uses the mailbox as a temporary storage until the mails are retrieved to the client, while in the IMAP protocol the mailbox is a permanent storage in a true client server implementation of mail.

Both the imap and the pop3 process use dovecot-auth as authentication facility, any client needs to authenticate by means of a valid username / password combination. Dovecot-auth turns to the mysql database where the usernames, passwords and other mailbox properties are stored.

Mail clients send mail by means of the Send Client mail port, which is realized through the smtpd process that also receives incoming mails from third party providers. It is the responsibility of the qmgr process to decide whether a mail needs to be forwarded to third party providers or whether the recipient is a local recipient. In the first case, qmgr hands the mail to the smtp process, otherwise to pipe as described above.

Squirrel is an implementation of a webmail gateway. Squirrel hooks into the web server apache. A web browser sends an HTTP request to an address at which the apache web server is listening for incoming requests. Incoming requests to a specific URL sub tree are deferred by apache to squirrel. Squirrel interprets the HTTP requests, accesses the mailbox and the contained mails through the imap process and responds by sending HTML formatted responses to the web browser.

In addition to providing the web mail access service, apache is the entry point for further web based services like user control, mail administration and the promotion on web.

The mysql database maintains tables of information that is used by other processes as the central source of truth for user accounts, mailboxes, filters, forwarding rules and their properties. These have already been identified as business entities in . For example, mysql provides information about a mailbox to the deliver process, i.e. the storage size assigned to a particular mailbox. The storage size is then enforced by the deliver process when it stores a mail in the filesystem directory that implements the actual mail storage for that particular mailbox.

User accounts and mailboxes are provisioned in the infrastructure by creating corresponding entries in the tables of mysql. Not all connections from mysql to other processes are shown in the diagram.

A named process implementing the DNS protocol for answering name service requests is also part of the provider infrastructure. In our simplified example there are no hosted mail domains and all users share the same mail domain (e.g. mail addresses have the form …@yahoo.com), therefore the DNS entries are static. In a more complex example, a mailbox might have associated mail domains (e.g. … @acme.de) that are announced as DNS MX records to the internet.

Note that this is still a simplified example, it is designed to be complex enough to be relevant, but simple enough to be presented in a paper.

### SID Model

An essential concept for the modeling of the mail infrastructure is the SID LogicalDevice, which is the equivalent of a process. The SID model () gives us the following hint in the description of the class LogicalDevice: “Conceptually, this represents the "brains" of the device”. This is similar to the UML blocks or components.

A SID LogicalDevice has a set of SID DeviceInterfaces that are similar to UML ports. A SID DeviceInterface can be addressed by a number of SID NetworkAddresses, this shows that the SID DeviceInterface has the character of a service access point. The SID protocol is a formal set of rules and conventions that governs how two entities exchange information, therefore this nicely maps to the UML interface.

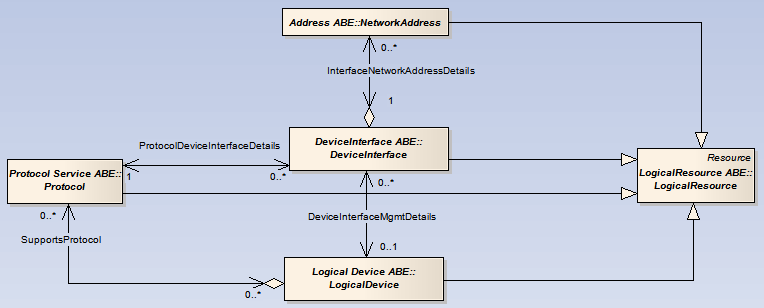


Figure : Essential SID classes for the modeling of processes as SID LogicalDevices

At the level of logical resources, the SID provides quite a large number of patterns and classes with descriptions and associations that leave room for semantic interpretation and different modeling styles. In fact two persons will probably never come up with the same model for the same technical arrangement. The SID classes that are presented here are a very small fragment of the SID that have been selected to highlight essential concepts on the level of logical resources.

In lack of a SID class that could be used to connect two SID DeviceInterfaces or two SID LogicalDevices in the style of an UML assembly connector ( ), we extend the SID with the class AssemblyConnector, as shown in the subsequent diagram.

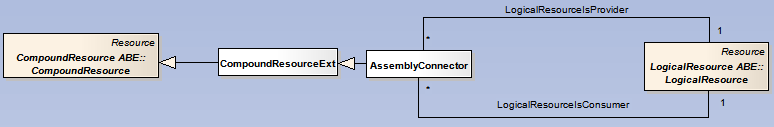


Figure : Extending the SID with an AssemblyConnector to connect LogicalResources

In the next example we study a subsection of the overall model of logical resources: The context of the imap and pop3 processes is shown in the following UML component diagram.

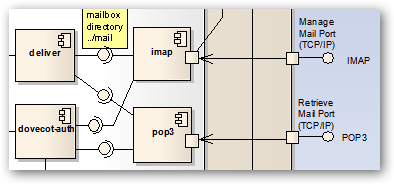


Figure : The context of the imap and pop3 processes

A straightforward SID representation of that subsection is shown in the following object diagram without bothering to model the UML delegate association.

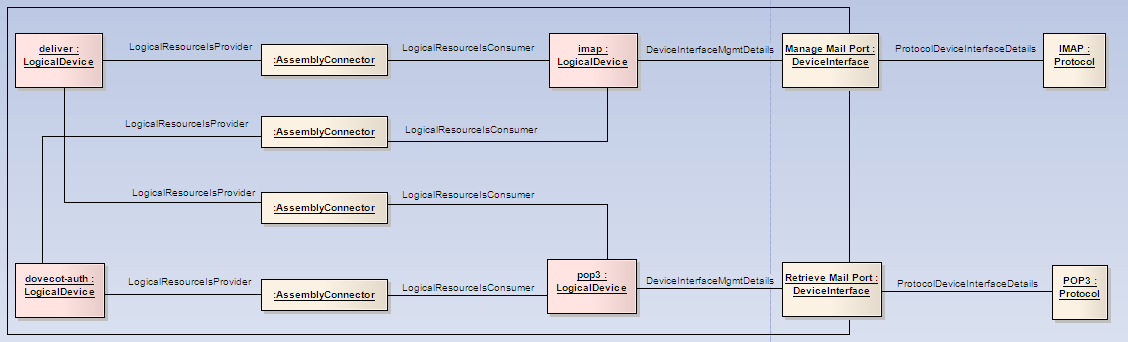


Figure : SID model of the context of the imap and pop3 processes

For the entities shown in the diagram there need to be corresponding specifications which are not shown because that wouldn’t add any new insights. A full-fledged model would also show the operating system, file systems and so on.

The following diagram shows the logical resources for the user account and the mailbox. They are hosted by the mysql logical device, which makes the user account and mailbox information available to the relevant processes of the infrastructure.

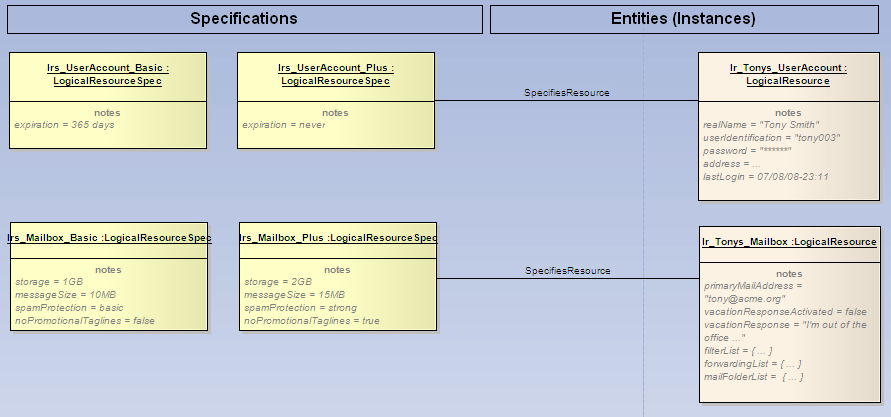


Figure : SID object model of user account and mailbox logical resources

### Requirements to Lower Layers

The logical resources are a source of requirements to the design of the physical resources. Compatibility requirements address the relationship between logical and physical resources, e.g. an operating system might not run on every hardware platform. Sizing requirements address the capacity issues of storage size, communication bandwidth and processing performance.

## Production of Resource-facing Services by Logical Resources

### Domain-specific Analysis

The following relationships between ports and exposed services have already been analysed ():

* Manage Mail Port exposes Manage Remote Mailbox RFS
* Retrieve Mail Port exposes Retrieve Mail RFS

Obviously it is the imap process that provides the Manage Mail Port, thus the imap process implements the resource-facing service Manage Remote Mailbox RFS exposed at the port. The pop3 process provides the Retrieve Mail Port, thus the pop3 process implements the resource-facing service Retrieve Mail RFS. This is derived from the diagrams in .

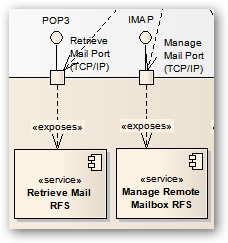
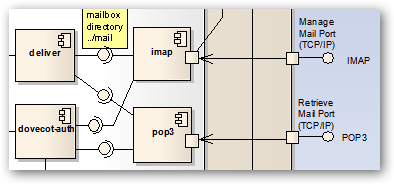
 

Figure : Relationship of resource-facing services and logical resources

### SID Model

shows the relationship of the resource-facing services to the underlying logical resources.

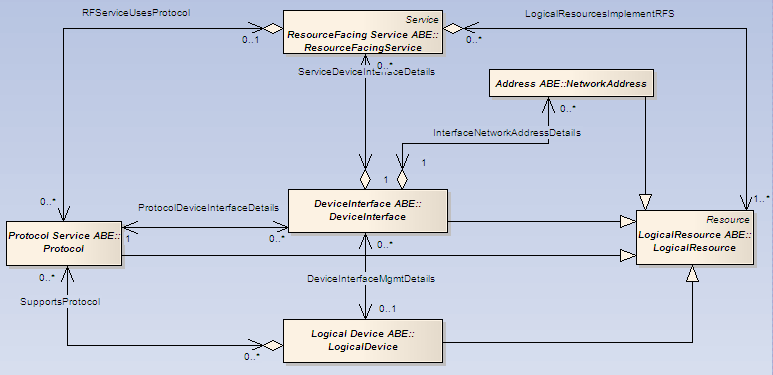


Figure : Selected SID classes highlighting the relationship between resource-facing service and logical device

The imap and pop3 process scenario can be modeled in a SID based object diagram as shown below (). Note that the resource-facing services are only modeled at the external system boundary of the infrastructure, in correspondence with the black-box models that have been developed in the previous sections.

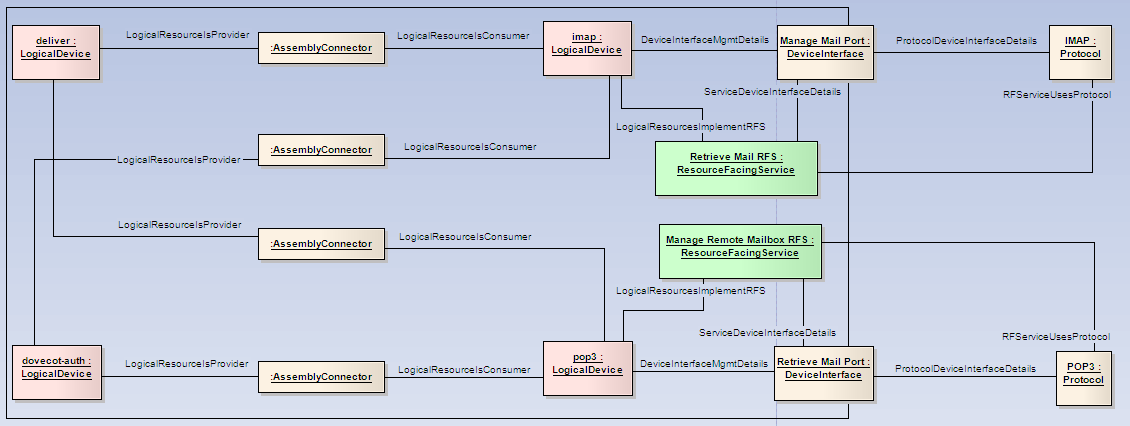


Figure : SID object model of the resource-facing services exposed by the imap and pop3 processes

The question arises whether the dovecot-auth process should be connected to the resource-facing service by means of a LogicalResourcesImplementRFS link: If the dovecot-auth isn’t running, the resource-facing services Retrieve Mail RFS and Manage Remote Mailbox RFS could not be accessed, because clients would not be authenticated.

We decided not to do that because the model is intended as a technical design which should be clearly separated from other interests like correlation in fault analysis. A good technical model will be an excellent basis to build additional views that cater for other interests like provisioning and fault analysis, but it is not a good idea to consider these aspects too early.

shows the mapping of resource-facing services to logical resources for the two wrapper resource-facing services user account and mailbox. The mapping is a 1:1 pass-through mapping. For each user account and mail box entity at the RFS level there is a corresponding entity at the logical resource level.

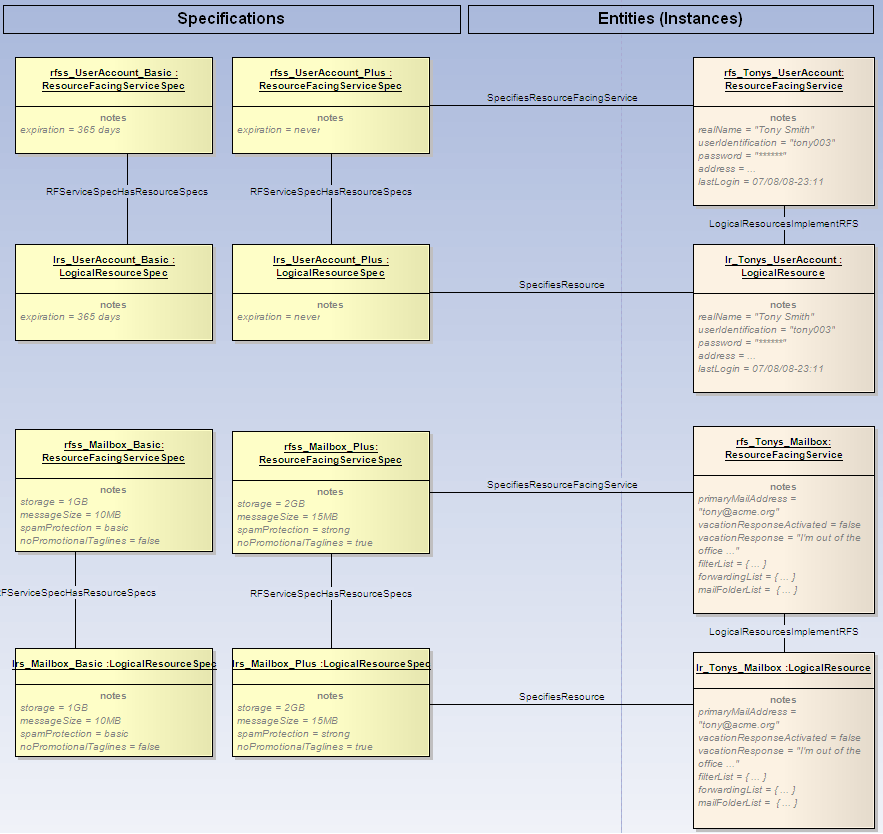


Figure : SID object model for the mapping of UserAccount and Mailbox to logical resources

## Physical Resources

### Domain-specific Analysis

At the physical level, the service provider infrastructure consists of a single mail server that is connected to a firewall by LAN, to keep our example simple (). The firewall protects the mail server against attacks from the internet. The firewall is connected to the internet via a router and a leased line.

The logical resources are a source of requirements to the design of the physical resources:

* compatibility requirements address the relationship between logical and physical resources, e.g. an operating system might not run on every hardware platform
* sizing requirements address the capacity issues of storage size, communication bandwidth and processing performance

Nevertheless, physical resources are quite independent from the logical resources that they host. For example, the mail server hardware can be replaced by a different model, and disks and memory can be added without impacting the structure of the logical resources.



Figure : Physical resources of the service provider infrastructure

Obviously up to now we didn’t consider the connectivity of the infrastructure to the internet in our models of customer- and resource-facing services. The reason is that we have concentrated on the services that directly contribute to the user perception, thus ignoring additional enablers that can be considered to provide services in their own right.

### SID Model

SID provides a set of classes to describe the physical composition of network equipment. These classes can be used as a framework for modeling the physics of the mail server, but need to be augmented by an additional class for the hard disk.

An overview of the SID model for hardware is shown in .

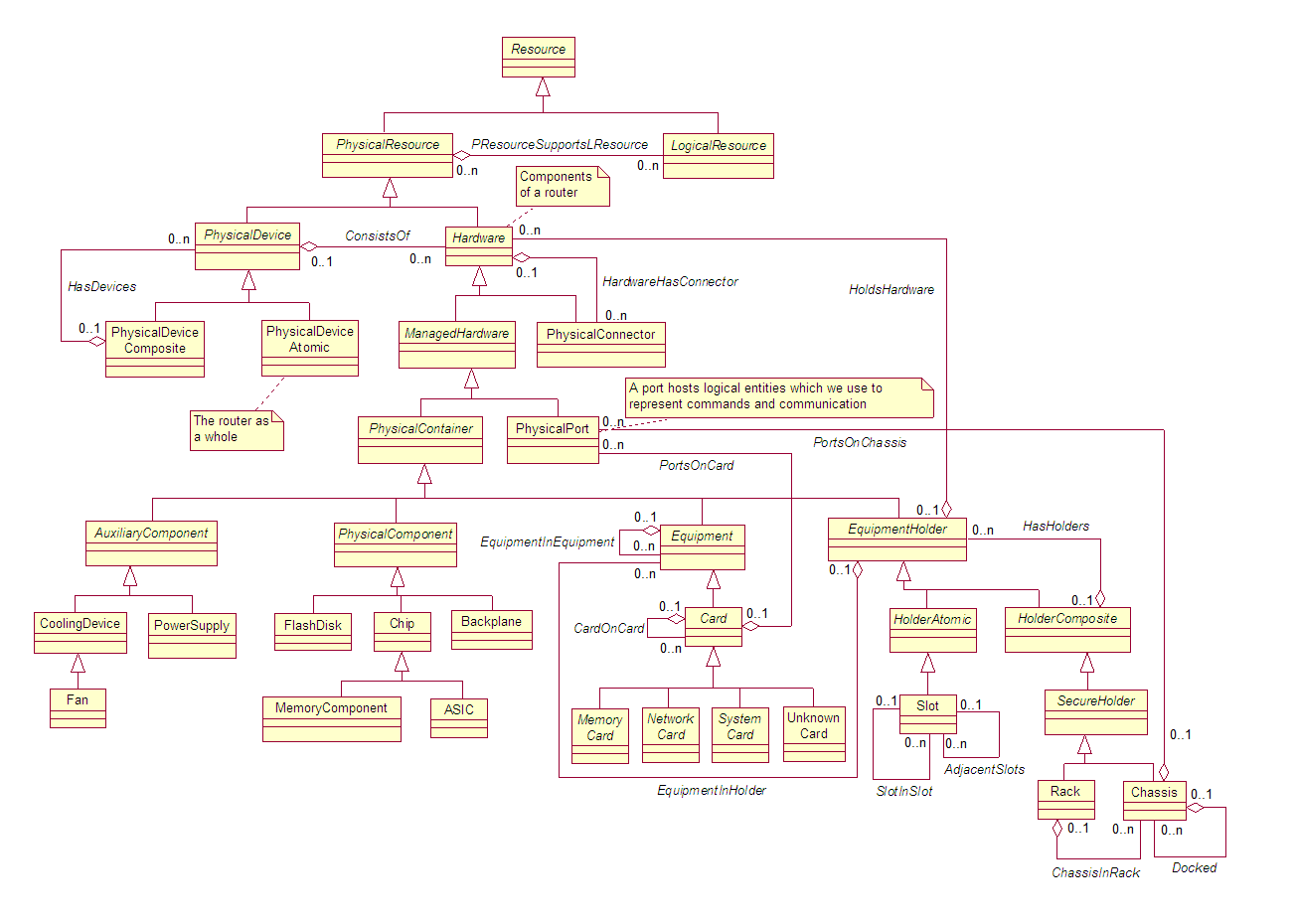


Figure : Selected SID classes to model hardware

Figure 36: Overview of SID model for hardware

sketches a SID model of the mail server’s physical resources. The entities are displayed on the

right side of the diagram, the specifications on the left side. The mail server has two identical hard disks, therefore both link to the same SID ResourceSpecification instance.

Note: The other ResourceSpecifications are not fully elaborated in the diagram.

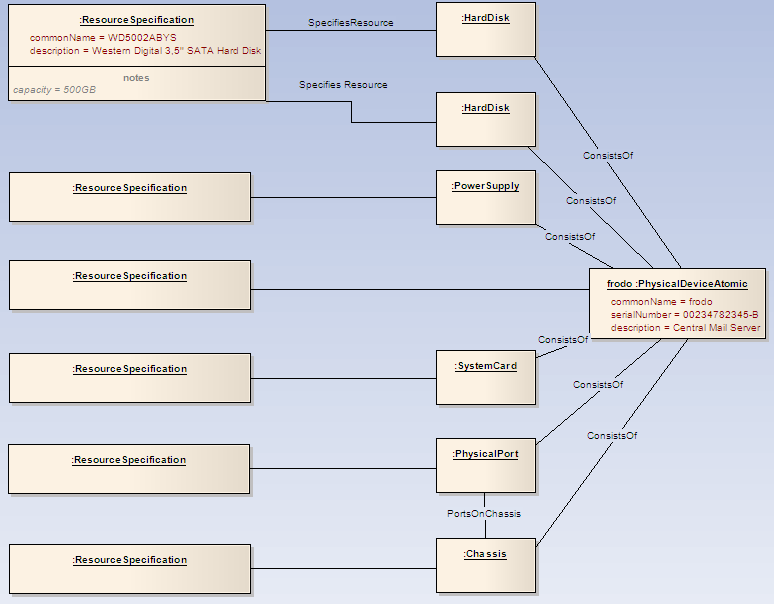


Figure : SID model of physical resources for the mail example

## Linking Logical and Physical Resources

### Domain-specific Analysis

Logical Resources need the support of physical resources; they cannot exist in free space: The processes need to be supported by computing hardware.

### SID Model

SID models the relationship between SID LogicalResources and SID PhysicalResources by the association PResourceSupportsLResource ().

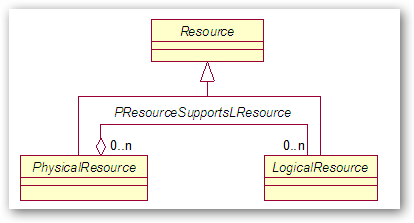


Figure :The link between physical and logical resources in the SID

This results in connecting all SID LogicalResources to their underlying SID PhysicalResources. In a simple model, all SID LogicalResources of our mail example would link to the SID PhysicalDeviceAtomic frodo that represents the mail server as a whole. This is sketched in for some of the processes only.

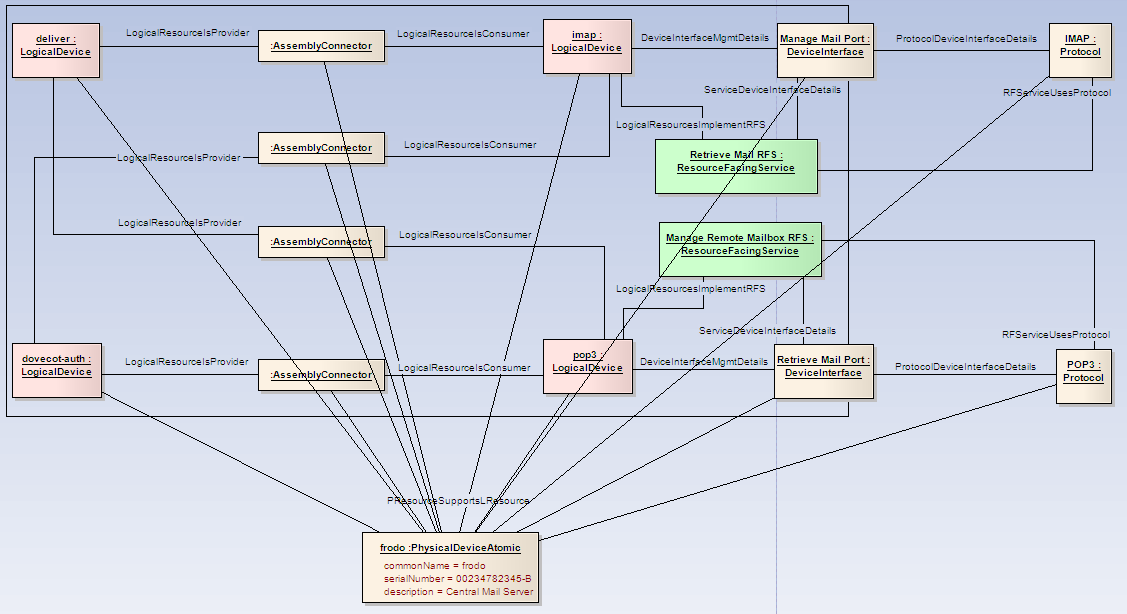


Figure : SID model of logical devices running on a physical device

# Summary

The example details the construction of an e-mail offering from the corresponding services and resources by means of a model.

The example has initially been created as input for the analysis of lifecycle aspects of the different artifacts by the PLM team. The analysis of these lifecycle aspects itself is not part of this study.

The methodology used in this study combines the strengths of UML component modeling and SID data modeling. UML component models are used at the service and resource level to illustrate the technical construction as business background for the development of the corresponding SID model.

The ability to express the construction of products in a ubiquitous model that is understood by a wide range of stake holders with different backgrounds brings significant benefits for the telecommunications and IT industry. A model similar to the one presented in this study is a valuable basis for the creation of catalogs and inventories that support lifecycle, provisioning and assurance processes – speeding up the product lifecycle and fostering the reuse of existing capabilities.

This requires models that at the same time

* are visually expressive and easy to understand
* are formal enough to have clear semantics
* reflect the specifics of the underlying technology
* address the viewpoints of all relevant stakeholders - from portfolio management to engineering departments and IT specialists

The methodology applied in the example provides a starting point to create such models and is based on standards like UML and SID.

Nevertheless, discussions within the team involved in creating the model, within the TMF community and experiences from similar projects show that today it is still a challenge to create a universal and unambiguous model of a particular product and its underlying technology.

4 Appendix 1 – Glossary

This glossary presents the definitions of key SID concepts in parallel with the more pragmatic viewpoint taken in the case study.

The pragmatic viewpoint is not intended to be normative, but rather shall give a starting point for a practical adoption of SID concepts for the construction of ICT products from services and resources.

| **SID IX definition** | **Pragmatic viewpoint  used in the case study** |
| --- | --- |
| **Product Specification** | |
| A detailed description of a tangible or intangible object made available externally in the form of a Product Offering to Customers or other Parties playing a PartyRole. A Product Specification may consist of other Product Specifications supplied together as a collection. Members of the collection may be offered in their own right. Product Specifications may also exist within groupings, such as Product Categories, Product Lines, and Product Types. | A product specification describes the value proposition, the functionality, the quality, the branding and all other aspects that are known to be important to the customers in a targeted market segment.  From the perspective of a customer, a product specification is an answer to the question “what will I get from a service provider and what is it good for?”, which is frequently detailed in a corresponding product brochure. It contains as much detail as needed for guiding the buying decision and for creating a legally valid agreement.  The same product specification can be offered at different prices to different market segments served by different sales channels – these aspects are therefore intentionally not part of the product specification.  A product specification typically carries the product name that is used for communication with customers, e.g. in marketing campaigns. |
| **Product Offering** | |
| The presentation of one or more Product Specifications to the marketplace for sale, rental, or lease for a Product Offering Price. A Product Offering may target one or more Market Segments, be included in one or more Product Catalog, presented in support of one or more Product Strategies, and made available in one or more Places. Product Offering may represent a simple offering of a single Product Specification or could represent a bundling of one or more other Product Offering. | From the perspective of a customer or employee browsing a product catalog, the product offering is an entry in the product catalog that consists of a product specification with its corresponding price tags (product offering prices).  From the viewpoint of a product manager, a product offering defines what functionality is provided at what price to which market segments over which sales channel.  The name of a product offering can best be thought of as <product name>\_sold\_to\_<market segment>\_through\_<sales channel>, where the <product name> comes from the underlying product specification. |
| **Product Catalog** | |
| A list of ProductOfferings for sale, with prices and illustrations, for example in book form or on the web. ProductCatalogs can be used by Customers during a self-care ordering process and may be used across one or more DistributionChannels. | Nothing to add. |
| **Product** | |
| A ProductOffering procured by a Customer, or other interested Party playing a PartyRole, appearing as a BusinessInteractionItem, which could take the form of an Agreement. ProductSpecificationCharacteristic(s) in part define the Product. A Product is realized as one or more Service(s) and/or Resource(s). | A product is the object of the mutual agreement between a service provider and a customer, thus realizing a product offering: A product (instance) documents the promise of a service provider to provide an instance of the functionality described in a product specification in return for the promise of payments according to the defined product prices.  Another useful real-world interpretation is that a product is a right granted to a customer to use specific services and resources.  A product cannot be used by a customer, because it is only a promise or a right of use. The user obtains the value described in the product specification only through services and resources which realize the promise.  A product has a commercial status that reflects the state of the agreement or right of use, like planned, designed, activated or disconnected. |
| **Customer Facing Service Specification** | |
| A Customer Facing ServiceSpec defines the properties (attributes) common to a particular Customer Facing Service used to realize the associated Product(s). This entity serves as a common basis to build any set of Customer Facing Services that the service provider needs. | A customer-facing service specification describes a functionality that the service provider provides to a customer and that the customer can access at a well-defined interaction point, which is typically realized on an end-user device like a mobile handset or a PC.  While the product specification only contains the information needed for a buying decision and a legally binding agreement, the customer-facing service specification describes the details of the functionality that can be accessed by a customer.  A customer-facing service specification hides the details of its underlying technical implementation as much as possible.  The name of a customer facing service specification refers to the functionality represented, e.g. Client Mail Access Service or Multimedia Messaging Service.  This narrow, device-centric interpretation can easily be extended to services provided by humans, e.g. hotline services. |
| **Customer Facing Service** | |
| A Customer Facing Service defines the properties of a particular related Service that represents a realization of a Product within an organization's infrastructure; This is in direct contrast to Resource Facing Services, which support the network/infrastructure facing part of the service. For example, a VPN is an example of a Customer Facing Service, while the sub-services that perform different types of routing between network devices making up the VPN are examples of Resource Facing Services.  Customer Facing Services are directly related to Products as well as to Resource Facing Services. Resource Facing Services are indirectly related to Products through the relationship between Product and Customer Facing Services. This enforces the relationship to Products while keeping Services that are not directly realized by Products (i.e., Resource Facing Services) separated from Products. | A customer-facing service is a customer-specific instance of a functionality that the service provider provides to a customer at a well-defined interaction point, which is typically realized on an end-user device like a mobile handset or a PC.  This represents the service that a customer can actually use and which is produced through the interaction of an end-user device with the resource-facing services provided by a service provider infrastructure.  A customer-facing service has an operational state that can be easily observed by the customer, e.g. operational, degraded or failed.  The functionality of a customer-facing service is described by a customer-facing service specification. |
| **Resource Facing Service Specification** | |
| This is an abstract base class for defining different types of Resource Facing Service Specs. A Resource Facing Service Spec is an abstraction that defines the invariant characteristics and behavior of a particular Resource Facing Service. The invariant portion serves as a single common basis to build a set of variable Resource Facing Services that all use this common Resource Facing ServiceSpec.  This class can be thought of as a template, which represents a generic specification for implementing a particular type of Service. A Service Specification may consist of other Service Specifications supplied together as a collection. Members of the collection may be offered individually or collectively. Service Specifications may also exist within groupings, such as within a Product. Service Specification inherits from Specification, which inherits from Managed Entity. | A resource-facing service specification is a description of a functionality that can be exposed at a device interface (a different name for a service access point) and made be available by a service provider’s infrastructure.  A resource-facing service specification hides its underlying implementation through hard- and software. |
| **Resource Facing Service** | |
| This is an abstract base class for Resource Facing Services. A Resource Facing Service is an abstraction that defines the characteristics and behavior of a particular Service that is not directly seen or purchased by the Customer. Resource Facing Services are “internal” Services that are required to support a Customer Facing Service. The Customer obtains Customer Facing Services via Products, and is not aware of the Resource Facing Services which support the Customer Facing Service(s) that is being obtained directly by the Customer via a Product. For example, a VPN is an example of a Customer Facing Service. This particular type of VPN may require BGP to support it. Customers don’t purchase BGP, and hopefully aren’t even aware that BGP is running. Therefore, BGP is an example of a Resource Facing Service.  Customer Facing Services are directly related to Products as well as to Resource Facing Services. Resource Facing Services are indirectly related to Products through the relationship between Product and Resource. This enforces the relationship to Products while keeping Services that are not directly obtainable via Products (i.e., Resource Facing Services) separated from Products. | A resource-facing service is a functionality that is exposed at a device interface (a different name for a service access point) that is identified by a concrete network address and which is made available by a service provider’s infrastructure.  This represents the service that an end-user device invokes over a well-defined protocol. Depending on the technology, a resource-facing service may serve many different end-user devices (shared service) or serve a single end-user device (dedicated service).  A resource-facing service has an operational state that indicates whether the functionality can successfully be invoked by an end-user device.  The functionality of a resource-facing service is described by a resource-facing service specification.  Note that the case study only considers resource-facing services at the external boundary of the infrastructure, and not the service interworking within the infrastructure. |
| **Logical Resource Specification** | |
| This is an abstract base class that is used to define the invariant characteristics and behavior (attributes, methods, constraints, and relationships) of a Logical Resource. | A logical resource specification is a description of the characteristics and behaviour of logical resources that are considered to be of the same kind. |
| **Logical Resource** | |
| This is an abstract base class for describing different logical aspects of devices (e.g., Device Interfaces) that constitute a Product. It has two main purposes: (1) to collect common attributes and relationships for all logical entities, and (2) to provide a convenient, single point where relationships with other managed objects can be defined. | A logical resource is a part of a service provider’s infrastructure that is immaterial and can’t be touched.  The logical resource’s characteristics and behaviour are described by its logical resource specification.  Logical resources are hosted by physical resources; they can’t live on their own. |
| **Logical Device** | |
| This is an abstract base class for representing logical concepts and services that can be managed that are associated with the device as a whole. This class represents a convenient aggregation point for combining different aspects of a device (e.g., software contained in the device, protocols that the devices runs, the set of services that it offers, and so forth). It also enables the device itself to have a single logical manifestation.   Conceptually, this represents the "brains" of the Device. For example, it represents the set of entities required for a Router to know how to route packets.   Please see the DEN-ng Resource model for more details. | A logical device is a logical resource which is able to perform communication, information processing and management tasks, typically in form of a software or firmware process.  The logical device implements the behaviour that is exposed through resource-facing services at the logical device’s device interfaces. |
| **Device Interface** | |
| This is a concrete class that represents the (logical) interface or sub-interface of a device. This is not a transmission entity; rather, Device Interfaces are used to program Services and Logical Resources on a Device. Put another way, Device Interfaces can be used to model interface-specific commands which are used to bind an appropriate set of TPs. | A device interface is a logical resource that serves as service access point of a logical device. The device interface exposes the behaviour implemented by the logical device (aka process) as a resource-facing service at a concrete network address over a well-defined protocol.  For example, it is used to program a logical connection from a device to a network medium. Different types of DeviceInterfaces exist for the different types of network media (e.g., IP vs. ATM) that are used in a network to enable such media to be programmed. The combination of a LogicalDevice and a DeviceInterface is what a developer programs to define Services that run on the device. In this case, a LogicalDevice serves to orchestrate the functionality of the set of DeviceInterfaces that are used.  Please note that this is an architecturally significant entity. In reality, it is a System/Implementation View entity, as its purpose is to administer the creation and configuration of other entities using one or more languages (e.g., CLI, SNMP, or TL1). Hence, it is defined in the business view, as it is used to provide a basic configuration pattern. Additional work in the system view will expand upon this pattern, and help focus the SID-MTNM/MTOSI harmonization activities. |
| **Network Address** | |
| This class represents the generic concept of a network address. Its subclasses define different types of addresses of different technologies, such as an IPAddress vs. an IPXAddress. Its utility lies in its ability to serve as a convenient point for sourcing and terminating relationships. This eliminates undue duplication of relationships that interact with the subclasses of NetworkAddress. | A network address identifies a device interface (aka service access point) as an endpoint for communication. |
| **Protocol** | |
| A Protocol is a formal set of rules and conventions that governs how two entities exchange information (usually over one or more types of network media).   This is an abstract base class for representing Protocols that can be managed. This class represents a convenient aggregation point for defining how Protocols are managed and used.   Please see the DEN-ng Service model for more details. | A Protocol is a formal set of rules and conventions that governs how two entities exchange information (usually over one or more types of network media). |
| **Physical Resource Specification** | |
| This is an abstract base class that is used to define the invariant characteristics and behavior (attributes, methods, constraints, and relationships) of a PhysicalResource. | A physical resource specification is a description of the characteristics and behaviour of physical resources that are considered to be of the same kind. |
| **Physical Resource** | |
| This is an abstract base class for describing different types of hardware that constitute a Product. It has two main purposes: (1) to collect common attributes and relationships for all hardware, and (2) to provide a convenient, single point where relationships with other managed objects can be defined. | A physical resource is a part of a service provider’s infrastructure that can be touched. |
| **Physical Device** | |
| This is an abstract base class for representing hardware devices that can be managed. This class represents a convenient aggregation point for combining different aspects of a device (e.g., its physical composition as well as the set of services that it offers). It also enables the device itself to have a physical manifestation. Examples of this class include routers and switches, computers, and other end-devices that are managed. | A physical device represents a hardware device as a whole, e.g. a router or a server.  A physical device may consist of further physical resources like hard disks, power supply, physical ports and chassis. |

5 Appendix 2 – The Relationship between Customers, Products and Customer-facing Services

From discussions about the example presented in this case study it became obvious that there are at least two viewpoints within the TMF community about the relationship of customers to products and customer-facing services.

The case study is not intended to be dogmatic about the semantic interpretation of the entities of the example, but presents a consistent and helpful viewpoint of the different concepts involved. There is ongoing work in the SID Customer/User subteam to create a differentiated model of customers and users, and their relationship to products and services.

The following table matches the viewpoint of the example with an alternative viewpoint that has been encountered in the discussions within the TMF community.

|  |  |
| --- | --- |
| Viewpoint of the example: User-consumes-service approach | Alternative viewpoint: User-consumes-product approach |
| * a product serves as a packaging of services by a provider that can be purchased, leased or rented by a customer * customers and providers commercially agree on the basis of products * products cannot be used because they are only packaging, e.g. you can't make a phone call with a product, but only with the service that realizes the product * a user uses a customer-facing service to obtain the value promise of the product, e.g. to make a phone call * the user experience of using a customer-facing service is created at an interface realized by an application in an end-user device * the application on the end-user device invokes resource-facing services by communicating over protocols with device interfaces (service access point) that constitute the boundary of the provider infrastructure * a user may administrate aspects that influence the behavior of the customer-facing service (like call-forwarding) but not the commercial agreement through characteristics on the customer-facing service or by using another customer-facing service | * a product serves as a window for customers and users into the world of the services and resources used to realize the product. * Customer/user always deal with products, unless the individual/organization they represent is playing the role of a service technician or some similar role * customers and providers commercially agree on the basis of products * a product is what a customer and/or user perceives is being used, e.g. to make a phone call * the user experience of using a product is created at an interface realized by an application in an end-user device\* * the application on the end-user device invokes resource-facing services by communicating over protocols with device interfaces (service access point) that constitute the boundary of the provider infrastructure * a customer-facing service represents a functionality at the boundary of the service provider infrastructure in a protocol-agnostic way, it groups a set of resource-facing services that together provide the necessary technical functionality * a user may administrate the aspects that influence the behavior of the product (like call-forwarding), this changes characteristics of the product   \*) another way would be e.g. to call a hotline to get services performed by humans |

A key difference is the system boundaries that define the scope of the model. The case-study viewpoint explores the perspective of the human user, therefore the customer-facing service is observed at the end-user device. The alternative viewpoint relates the customer-facing service directly to the service provider infrastructure, the concept of end-user device is not considered here.

As far as the resulting SID model is concerned, there is no big difference between the two viewpoints, but the UML component models shown in “” and “” will be fundamentally different when assuming the alternative viewpoint.

6 Administrative Appendix

## 6.1 Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Date Modified** | **Modified by:** | **Description of changes** |
| 0.2 | 15 September 2008 | Martin Gutzki Michel Besson | Created |
| 0.3 | 25 September 2008 | Martin Gutzki | Redesign of the SID example and text in order to reflect a strictly hierarchical decomposition, thus eliminating the link from products to logical resources as modeled in version 0.2 |
| 0.4 | 12 November 2008 | Martin Gutzki | Compliance with Yahoo’s copyright terms for screenshots and glossary added, minor reformatting |
| 0.5 | 15 December 2008 | Martin Gutzki Michel Besson | Updated according to comments and discussions |
| 0.6 | 13 May 2009 | Martin Gutzki | Figure 1 and related text added, minor editing |
| 0.7 | 19 May 2009 | Michel Besson | Minor editing changes |
| 1.0 | 23 June 2009 | Michel Besson | Minor editing changes; ready for ME review. |
| 1.1 | 25 June 2009 | Alicja Kawecki | Minor updates for web posting and ME review |
| 1.2 | 18 February 2010 | Alicja Kawecki | Updated to reflect TM Forum Approved status |
| 1.3 | 26 April 2010 | Martin Gutzki and Raja Sekhar Ravi | Included modeling approach and documentation to address the CR-3 raised by John Reilly. This is to represent views of Product-Service golden nugget provided by John Reilly. |

## 6.2 Company Contact Details

|  |  |
| --- | --- |
| **Company** | **Team Member Representative** |
| Deutsche Telekom | *Gutzki, Martin*  *M.Gutzki@telekom.de* |
| TM Forum | *Tina O Sullivan*  *tosullivan@tmforum.org* |

## 6.3 Acknowledgments

This document was prepared by the members of the TeleManagement Forum Product Lifecycle team:

Nektarios Georgalas, BT, Editor

Michel Besson, Amdocs  
Martin Gutzki (editor), Deutsche Telekom

Raja Sekhar Ravi (editor), Infosys

Kevin Morris, Convergys

Elaine Haher, Telcordia

Atit Patel, SquareHoop Ltd

Mark Watson, SquareHoop Ltd

John Wilmes, Progress Software

Tapani Honkanen, TeliaSonera

Jenny Huang AT&T

1. See <http://en.wikipedia.org/wiki/Marketing_mix> for reference [↑](#footnote-ref-1)
2. The viewpoint that customers in a user role interact with customer-facing services is not universally shared within the TMF community, see discussion in Appendix 2 [↑](#footnote-ref-2)
3. Please note that it is one possible proposal, but other service designers may have different proposals. In particular, the service designer has to take into account the existing situation. In the real world, the CFSSpecs already existing prior to the creation of a new Product must be considered to be reused. [↑](#footnote-ref-3)
4. Definition according to the INCOSE systems engineering handbook, [↑](#footnote-ref-4)
5. Definition taken from the OMG SysML specification [↑](#footnote-ref-5)