Service Theory

Communications services

We saw that Vail introduced the notion of a communications **service**
and that today we pay mostly for **Q**uality **of** **S**ervice

Classic communications services are pure connectivity services, defined by :

- endpoints
- bandwidth (or equivalent)
- delay (and possibly delay variation)
- priorities, access permissions, etc.

Modern communications services have non-trivial network functionalities, e.g.,

- firewall, IDS
- authentication, integrity, encryption
- performance acceleration,
- caching, CDN
- performance monitoring, application visibility

It is useful to consider such constructs in a *service theoretic* context

Services

What is a service ?

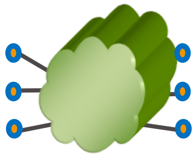
In economics a **commodity** is a marketable good, i.e., an item

- that provides utility
- that someone wishes to sell
- that someone else wishes to buy

Historically commodities were classified as either **products** or **services**

In order to provide communications we need both

- products (e.g., switches, routers, middleboxes, transceivers, etc.)
  and
- services (e.g., telephone/cellular, VPN, Internet access, etc.)



We call entities that sell products communications equipment **vendors**
and those who sell services communications service **providers**
We call entities that buy products **customers** or **consumers**
and those who buy services **users** or **subscribers**

Service description

Products may be formally described, e.g., in brochures and datasheets
but importantly potential customers can *see* and *feel* products

Formal description is more important for services
since they can’t be seen
and it is not always practical to try out a service before ordering

Formal languages, such as

- **W**eb **S**ervices **D**escription **L**anguage (W3C)
- **U**nified **S**ervice **D**escription **L**anguage (W3C, Theseus-Texo, SAP)
- Linked UDSL (W3C, KMI, SAP)

are advantageous for many service types
and enable delivery and trading of services over the Internet

Multi-provider services

Often no single service provider can deliver the entire service

In such cases service providers form partnerships

For example, international mail requires multiple postal services

In telecommunications we often speak of an **end-to-end service**
provided by multiple service providers

This can be accomplished in two distinct fashions:

- one service provider is the lead service provider, who :
  takes responsibility for the end-to-end service
  interfaces with the customer (service monitoring, billing, customer support)
  contracts out portions of the service to other service providers
- the customer deals directly with multiple service providers

The product–service spectrum

It was once popular to emphasize the *dichotomy* between products (such as routers) and services (such as a VPN)

Selling a tangible product was mostly a prompt one-time event

On the other hand, delivering a service involves a *life-cycle*

- negotiating service terms
- service planning, creation, provisioning, testing
- service monitoring, assurance and customer support
- billing
- service termination

The period from initial request to service fulfillment could take weeks

Today the distinction between a product and a service has blurred and economists talk about a product-service *spectrum*

Services often require deployment of products and product delivery generally involves service delivery

Consequences of product-service blurring

With the blurring of the product/service distinction

customers start expecting service attributes to be similar to product ones

Customers expect :

- service description to be as understandable as a product name

  if it did everything I wanted last time, it should this time
- service delivery to be as fast as product delivery

  click a button and instantly receive service
- service reliability to be similar to product quality

  perfect out-of-the-box, or return it

Unfortunately, these expectations are hard to meet with present technologies

In fact they are largely contradictory

- advanced services require longer commissioning times
- fast set-up may lead to low reliability or performance
- simple services are usually more reliable than complex ones

Slow service delivery paradox

Let’s consider service delivery (fulfilment) times

There seems to be an paradox :

- customers want instant gratification
- Service Providers desire faster *order-to-cash* cycles

If both sides want speedy delivery

why is new service delivery still slow ?

The paradox can not be explained away by physical constraints

since even services without installation (*truck rolls*) requirements

are characterized by slow delivery

This paradox has recently become painfully evident

as software products are now quickly developed

while new communications service types take months or years to deploy

Poor service in general

There are basically 4 reasons for poor service

all of which derive from the very nature of services

(in contradistinction to products)

- services are **intangible**
- services are **abstract**
- services are **nonexclusive**
- services are **commitments**

Services are intangible

Due to **intangibility** :

- Services can not be manufactured and stored
  - they must be created on-demand one-at-a-time
- Services tend to have many free parameters
  - there may be 5 kinds of coffee available in a café
    - but even a simple existing communications service template
    - can have 5 fields each with >10 values each
  - new service types require description of completely new constructs

Intangibility translates into lengthy set-up times, since :

- service parameters must be negotiated between customer and SP
- service parameters must be translated into technical parameters
- technical parameters feed complex optimization procedures (planning)
- the network then needs to be configured accordingly
- the configuration must be meticulously tested

Services are abstract

Due to **abstractness** :

- Services can be implemented in many different ways
  the only restriction being physical localities
- Variant implementations needs to be considered
  some sort of optimization carried out
- Different vendors usually have slightly different implementations
  and even different equipment from the same vendor
  may have different characteristics (CLI, state database, etc.)

Abstractness translates into lengthy set-up/modification and erratic service :

- scripts need to be written and maintained for all equipment types
- functionalities may not be available where needed
- functionality idiosyncrasies vary
- functionalities can not be readily migrated between equipment types

Services are nonexclusive

Once a product is delivered to a customer, no other customer can use it
but services share common infrastructures
and frequently compete for *collectively exhaustive* resources

Due to **non-exclusivity** :

- mutual interference needs to be carefully handled
  so as not to impair existing services
  and enable new services to conform to guarantees
- services need exhaustive joint activation testing

Non-exclusivity translates into lengthy set-up/modification and erratic service :

- existing service loads need to be tracked
- new services need to be verified and resource optimized
- momentary load peaks can cause nonconformance and information loss
- services are often prioritized (differentiated) rather than guaranteed
- network failures can lead to resource starvation of *fully protected* services

Services are commitments

Unlike products which are delivered in one-time events
services create obligations over protracted periods of time

For this reason

- customers and providers negotiate binding Service Level Agreements that define QoS KPIs that need to be monitored and maintained
- customers pay according to SLA level (*paying for QoS*)
- service providers incur penalties for SLA nonconformance

Before service delivery a service provider needs to ensure
that conformance is not only possible, but economically feasible

Thus, lengthy service activation testing is performed

- ITU-T M.2110 mandates 15 minutes, 2 hours or 24 hours of testing
- Y.1564 recommends
  - 2 hour test duration for single provider services
  - 24 hour test duration for multiple provider services

In addition, OAM must be continually run
in order to monitor SLA **commitments** throughout the service life-cycle

Research topics

- Services/products must be priced
  > production/maintenance cost
  ≤ what customers are willing to pay
  ≤ what competitors can offer comparative service/product
  How should a service/product be priced ?
  What is the strategy if pricing can continually change ?
  How can services be auctioned ?

- Service providers divide up their limited resources among customers
  - How can revenue be optimized ?
  - How can this optimization be done in “on-line” mode ?
  - how can oversubscription with penalties be optimized ?