INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19-21, 2003

AN INTEGRATED ARCHITECTURE FOR FUNCTIONAL PRODUCTS

Dr. Gunilla Sivard

Abstract

There is an increasing trend in industry of selling services in addition to physical products. In functional sales, companies sell the service of delivering the function of the product as opposed to selling the product itself. While there is a long tradition of structured development and management of purely physical goods, there is a lack of such methods for services, and for the mixture of service and goods - so called functional products.

This paper describes an effort of defining a formal model of services, contributing to the efforts of structuring, visualizing and managing functional products in analogue with purely physical products.

A modular architecture for services is presented, based on the paradigm of defining services as discrete, decomposable activities. Sub activities, so called service modules, are described in terms of the activity's properties and interfaces to other service modules. Since the method is based on principles for uncoupled design, it provides support for creating transparent and flexible architectures with a minimum of built in dependencies between functions and their physical implementation.

The service module aggregation and representation principles are described in relation to an industrial distribution case.

Keywords: Service, functional product, model, module, interface, architecture.

1. Introduction

1.1 Rationale for representing service

In industry of today, the development of services, rather than of purely physical products, constitutes a large part of a company's expansion [1]. Companies deliver a mixture of hardware, software and service, where services could be part of the final system, such as the telephone line switching performed by operators in the early days of the telephone, or be the activities of developing and maintaining the physical goods.

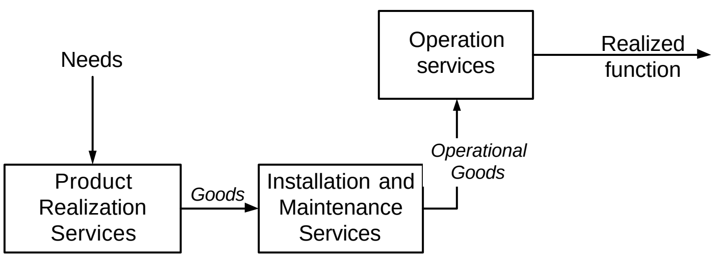

Figure 1 Services in relation to the goods, described in an IDEF0 model [8] (see section 3.1)

Services pertaining to the planning and design of systems are traditionally part of the sales activities and integrated with the price of the product. To a growing extent, though, these are separated out as individually priced and defined consulting products in themselves. For example, services related to a large system installation are in industry commonly divided into activities related to the system value chain [1]:

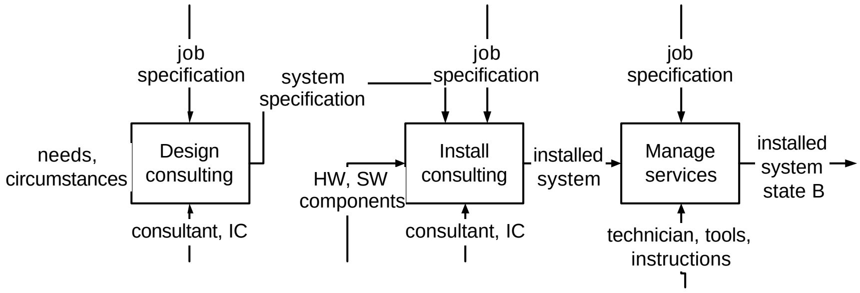

Figure 2 Services related to a system realization

Moreover, the trend is for many companies to turn to "Functional Sales" [2] as a more beneficial way of meeting the required function of their customers, functional sales meaning that the company instead of delivering a physical product, rather takes the responsibility of delivering the functionality of the product, often with an addition of other services required to meet the needs of the customer. This type of business thus means delivering a service, rather than delivering a product, for example delivering the service of "drying material" rather than selling an industrial fan. Apart from the changes to the organization and sales strategies of the company, this requires means for an efficient management and reuse of service descriptions and of the resources required to perform the services.

Design efforts will thus have to address the design of services that add value to and utilize parts as well as of the physical parts themselves, in order to deliver a whole function. Various means and standards for representing physical product information have been developed over the years [3], but there is a lack of methods for efficient development and management of services. Thus, there is a need for a method for structuring and describing services, as well as for structuring the mixture of service and goods.

1.2 Approach

The goals of this research are to support the development and management of functional products by: 1) defining how to structure services for efficient reuse and customization, and 2) defining how to model service information to support development and maintenance and finally 3) defining the relation between services and goods in relation to the overall function, in order to model a functional product.

For the structuring, a modular solution architecture is proposed. The approach is to apply principles for functionally decoupled design [4] and modularization [5][6], to structure a service in terms of independent, combinable service modules. The approach is based on the hypothesis that a service is a discrete process that, in analogue with a physical product, can be decomposed into a structure of components with defined functions and interfaces. Such modules could be controlled, varied and combined in various ways to create a large set of solution alternatives. A model of service is proposed, describing a service as a value adding activity with function, interfaces and properties - properties that in turn are materialized by the resources, which are used to perform the activity. A modular architecture is applied to an industrial service case study, based on existing principles for modular products and information modeling of product families with variants [7].

1.3 Definitions and limitations

There are several ways of defining concepts related to the selling of functions. In this context, the term *functional product* refers to solutions that contain a mixture of services and goods. These solutions may be sold directly to a customer, or may be parts of realizing the function in a functional sale. Since the focus of interest is on the representation of these mixed solutions, issues of different business strategies concerning sales, ownership of the physical hardware etc., are not considered.

2. Introduction to distribution example

An example will be used to clarify the representation of service, based on a case study at a large Swedish distribution company. Currently, their distribution is based on a number of integrated processes, each optimised for a specific type of delivery service. Although this is a pure service example, it serves the purpose of describing how to represent a service, whether it is a pure service or a service related to a physical product.

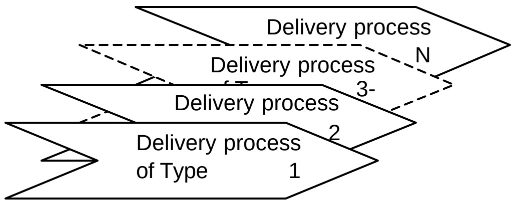

Figure 3 Integrated processes for each delivery

The company wants to split these processes into generic sub-activities that can be cost-measured, varied and combined to create customized processes.

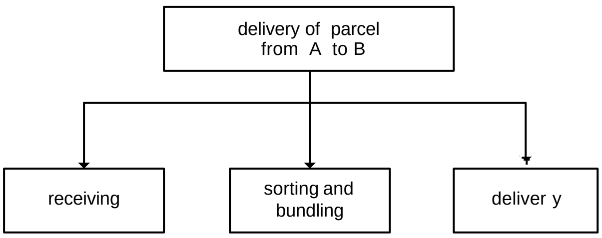

Figure 4 Main generic activities in distribution

3. Representing service

3.1 Defining service as an activity

A service realizes functions through an activity, as opposed to through an object. Assuming that this activity can be made discrete, it can be decomposed into a set of sub-activities. An activity is consumed in the same instance that it is created, which differs from physical objects, which can be (mass-) produced in advance, and stored for later consumption. What can be produced beforehand and stored for services, are the mechanisms for realizing it: tools, material, competence, information and procedures that can be designed for efficient service realization and reused again and again.

3.2 Modeling service in an activity model

Basically, an activity changes the state of matters, and in an IDEF0 model [8], an activity is modeled as a transformation of inputs into outputs by use of a resource mechanism, and controlled by a control-input. An input is always changed by the activity while the control is not changed; it just controls the way the activity is performed.

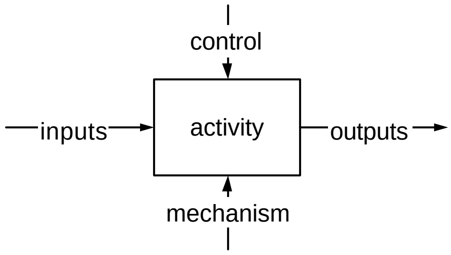

Figure 5 IDEF0 activity model

A service, being an activity, could be modelled in the same way. As with all models, a purpose, viewpoint and detailing level should be defined [9]. Purpose of this service model is to identify information related to the service, specifically identifying the reusable mechanism, the service function, the interfaces to other service modules, and relations to the hardware of the solution. Our viewpoint is the provider of the service, and the level of detail is low.
For this purpose, the mechanism constitutes people, machines, and tools, but also the supplier specific documentation of how a service is to be performed, such as instructions, methods and documented experience – the information resources. The in- and outputs describe the resources that are transformed by the activity, and states of the object (goods, humans) being "serviced": the "service object".

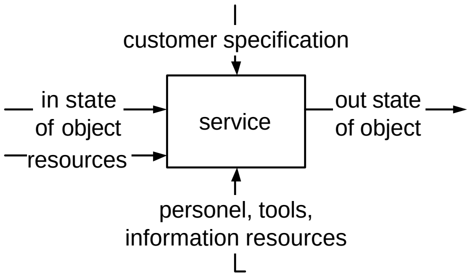

Figure 6 Activity model of service

3.3 Representing service as a module

A component is "a constituent part (of a whole)" [10], often described by its attributes and, when composite, by its structure of interconnected "subparts".

A module is a component with defined interfaces, function and properties describing what it achieves as opposed to how this is physically done. Main purpose of a modular architecture is to facilitate the creation of several product variants based on alternative modules. Products may vary "in space" as alternative variants in a product family, or "in time" as the new generations of an existing product. This interchange ability assumes that modules are more or less functionally independent, and that the physical interfaces are equivalent for different module alternatives. With the interface and properties remaining the same, the physical realization within the module may change without affecting the overall properties of the product. A service module is thus an activity that is defined in terms of its function, properties and interfaces.

3.4 Functional decomposition

A crucial step towards achieving functional independence between modules is to design the solution such that each functional requirement is independently realized in the solution. The theory of Axiomatic Design [4] defines axioms and design rules for designing such un- or decoupled solutions. The domain of design is described in terms of four interrelated domains: the customer, functional, physical and process domains.

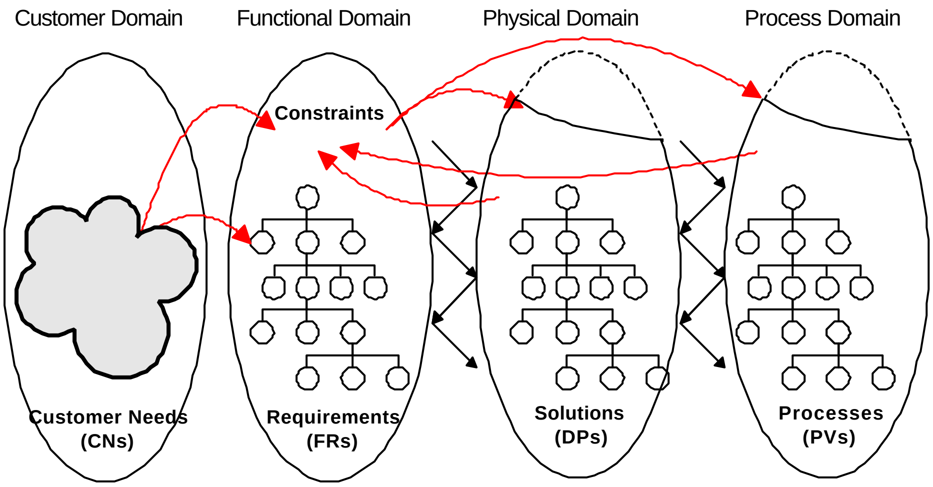

Figure 7 The domains of design

Customer needs are translated to functional requirements (FR) and constraints in the functional domain. FRs are realized by individual design parameters (DP) in the physical domain, and the constraints define requirements that cannot be satisfied by a single DP, such as e.g. cost or weight. The process of designing a solution is described as a zigzagging between requirements and design-, or solution- parameters, where each FR is realized by one DP, which in turn requires new FRs on the next level. This zigzagging between domains result in a functional decomposition tree structure:

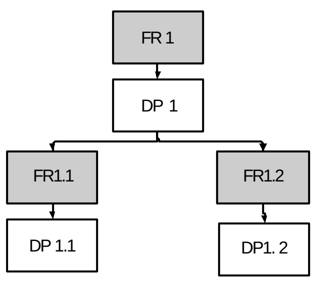

Figure 8 Functional decomposition

When designing the processes that realize the design parameters, a corresponding zigzagging between DPs and Process Variables (PVs) is performed.

It is important to understand that DP:s correspond to parameters of a solution, not to physical components. Further, it is crucial to *"Maintain the independence of Functional*

Requirements" (Axiom 1) by identifying and trying to minimize the dependence between one design parameter and other functions than the one it is supposed to realize.

In a functional decomposition of services, the design parameters correspond to principles for how the service activity is performed. The following structure visualizes a functional decomposition of the distribution example; grey boxes represent functional requirements, white boxes represent service design parameters.

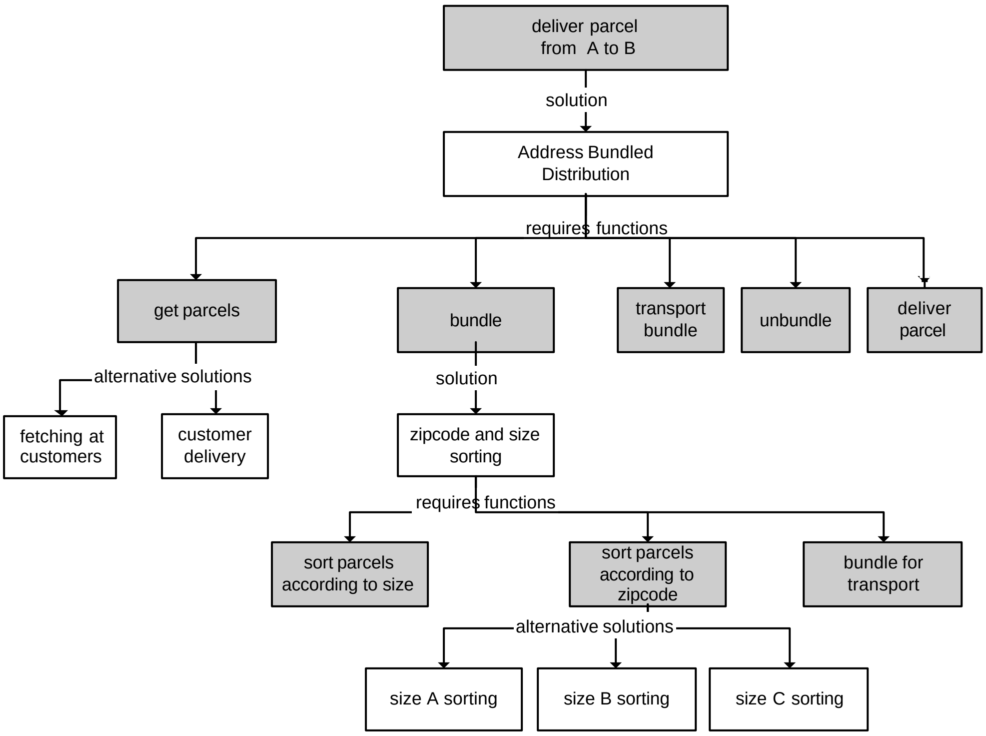

Figure 9 Functional decomposition of distribution service

3.5 Interfaces: Physical integration of service parameters

Solution parameters for the sub-functions of one main function may or may not be part of the same physically integrated solution; the integration is a separate issue of aggregating parameters into physical components.

One service module may realize only one function and materialize only one corresponding design parameter, but in analogue with physical objects, several design parameters are often integrated into one module.

For modules, it is paramount that interfaces can remain intact, and that the interface of all variants of a service-module matches the interface of the connected service-module. Thus, when integrating parameters that vary, all variants of how the service is performed should have the same interfaces.

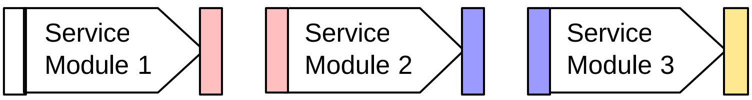

Figure 10 Connected service modules

Interfaces of a service module correspond to aspects of the activity that are relevant to the surroundings, and the interface information can be described in terms of the mechanism, in- and outputs of the activity model:

- Prerequisites to perform the activity
  - In-states that the service can handle:
    - Required attributes of the service object.
    - Location, time, etc. when service can be active
  - Information and resources needed to perform the action
- Physical interfaces of the resources performing the action
- Description of the result
  - Description of out-state:
    - Resulting attributes of the service object.
    - Location, time, etc. when service is completed
  - Information and resources resulting from performing the action

In the distribution case, one natural choice of service module aggregation is to create a "get parcels" module class, with alternatives "fetch at customer" and "customer delivery", both with an interface "unsorted parcels at location A" to the next service module class "bundle":

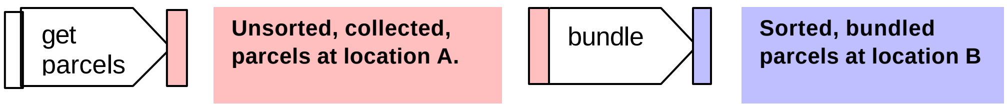

Figure 11 One module aggregation of service parameters

An aggregation can be made in many ways depending on the purpose and advantages with different ways of combining solutions. In the distribution case, it turns out to be more efficient to perform the size-sorting activity during the pick up transport and create the module "collect", which includes the size-sorting function as well as the 'get parcels' function.

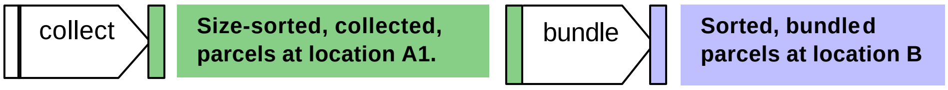

Figure 12 Alternative module aggregation

In this aggregation, solution parameters for different functional requirements are integrated in the same physical module, e.g. one module meets several functions.

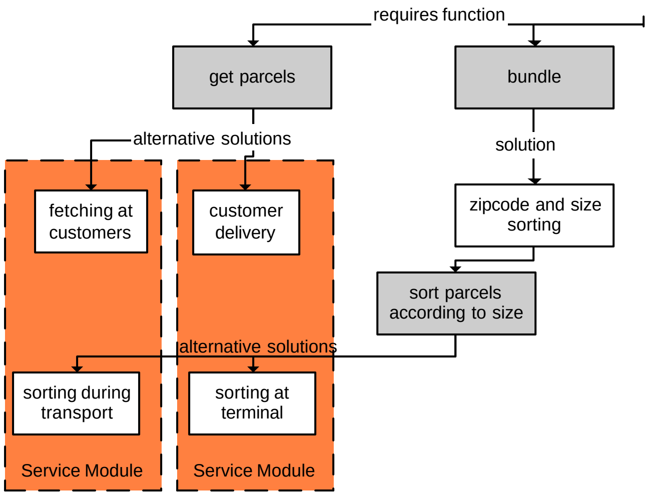

Figure 13 Aggregation of design solutions into service modules of distribution case

The integration of functionally independent parameters into the same physical component is not in conflict with the independence axiom of axiomatic design. From a modular perspective, though, alternative modules should have the same functional properties and interfaces – “points of hand over”.

3.6 Attributes and properties of service modules

The main function of an activity is turning an input into an output; typically the activity name and function are equal. Functional properties of the service activity, describing its abilities, quality, time etc. are not explicit in the IDEF0 model but are expressed in terms of the possible in/outputs and functional properties of the resources.

Physical attributes of an activity, describing how it is performed, are not explicit in the IDEF0 model either, but expressed in terms of the physical attributes of mechanism/resources used. The representation of resources, attributes and properties is not addressed further in this paper.

4. Architecture of functional products

A functional product denotes a solution that contains a mixture of services and goods. The purpose of describing a functional product is a) to be able to communicate the purpose of and interrelations between goods and services and b) to describe how it is to be efficiently realized. Services either use the goods as a resource for realizing a function, or transform some aspect of the goods.

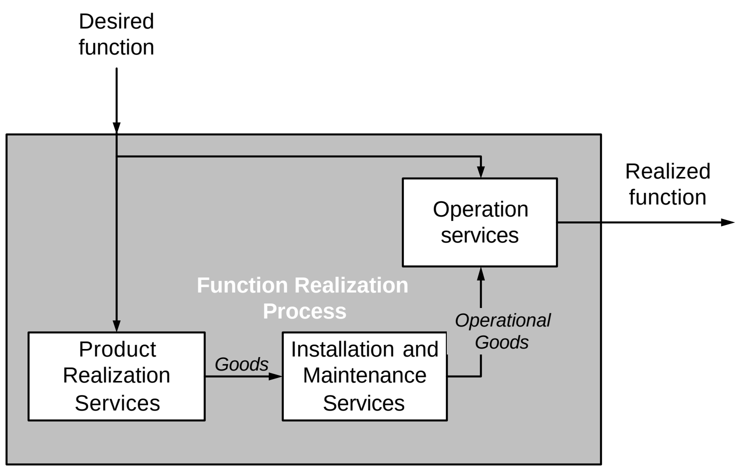

Figure 14 Relations between goods and service in the process of realizing function

The architecture of a functional product describes how activities and/or objects contribute to physically realize the desired function and properties. Design parameters represent physical solutions in a generic manner, thus it is assumed that design parameters of the goods and of the services can be entities in the same structure:

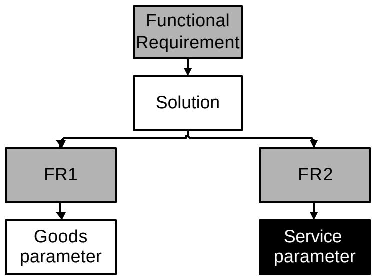

Figure 15 Integrated structure

To exemplify this, consider the functional product "provide data processing", i.e. a working computer system. Such a product needs to provide a solution to the functional requirement "repair system" which, depending on the technology used, may be realized automatically by a self-repairing processor, or by a traditional service personnel-led repair.

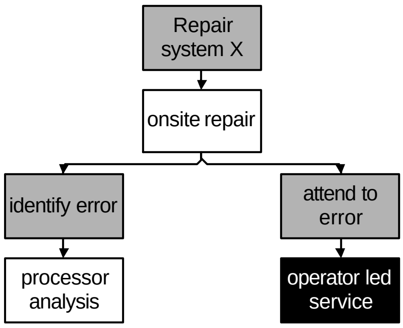

Figure 16 Example of structure of functional product

Thus, both the system solution, ultimately realized by software and integrated circuits, and the service, ultimately realized by human actions, can be parts of the same functional structure of activity solutions. Thus the realization of function by goods or service can be traced, the same rules of design be followed and goods and service solutions be managed in integration.

5. Summary – contribution

This paper describes an initial effort, in a new field of research, of representing service and functional products.

A modular service architecture is outlined, representing services in terms of combinable activity modules with specified function and interfaces. It is argued that modular service architectures will provide the same advantages concerning flexible development as modular product architectures, e.g. modules can be developed independently as long as the interfaces are kept intact. Further, such a subdivision makes the management of services more efficient by enabling e.g. the tracing of cost and the configuration of customer specific services.

A structure is suggested for functional products, which in an integrated way represents the decomposition of a functional product into goods and services. Since the structure is based on the axiomatic theory of design, it provides support for creating flexible architectures with a minimum of built-in functional coupling. Further, by stressing that functional requirements are independent of specific solutions, a premature locking into particular solutions is avoided, opening up the possibility of creating a flexible mixture of goods and services.

References

[1] Sivard G., "Modularisering av tjänster för effektiv kundanpassning", Study report, Royal Institute of Technology, Stockholm 2003

[2] Ölundh G. "Environmental and developmental perspectives of functional sales", Licentiate Thesis, Royal Institute of Technology, Stockholm 2003

[3] F.-L. Krause et al. "Product Modeling", Keynote paper, Annals of the CIRP Vol. 42/2 1993

[4] Nam P. Suh; "The Principles of Design"; Oxford University Press, 1990.

[5] M.M. Andreasen, *"Reduction of the complexity of product modeling by modularization"*, Conference on Produktmodeller, Linköping, Sweden 1998

[6] G. Erixon, 'Modular Function Deployment - A method for product modularization', Doctoral Thesis ISSN 1104-2141 Royal institute of Technology 1998.

[7] Sivard G., "A Generic Information Platform for Product Families", Doctoral Thesis, Royal Institute of Technology, Stockholm 2001

[8] NIST standard for function modelling, http://www.idef.com/idef0.html

[9] D. Ross, "Structured Analysis (SA): A Language for Communicating Ideas", IEEE Transactions on Software Engineering, vol. SE-3, no1, 1977

[10] Merriam-Webster OnLine: WWWebster Dictionary. 2003. Http://www.m-w.com/dictionary.htm