New Telco

Practices for transforming traditonal Telcos to an Internet Protocol world

made by

Product Foundry

Legal Notice

http://www.patterndeck.com/

Contact Information:

copyright@product-foundry.com
Product Foundry, Haarleemerweg 10,
1014 BE Amsterdam, The Netherlands

INTRODUCTION

Incorporating changes in our individual lifestyle is difficult, whether it be learning new skills or changing our habits. Incorporating meaningful change in large enterprises with several individuals is exponentially more difficult. It isn't a single solution applied by a single individual that changes a large and complex enterprise, but several solutions applied simultaneously and coherently by several individuals. Given that products unify an enterprise — in that individuals collectively design, produce, sell, support, operate, and decommission them — we offer in this article a set of solutions to transform enterprises on the basis of products. Pattern languages, which were originally conceived by Christopher Alexander as a means of describing effective design practices within architecture, can be used to represent the knowledge required for product lifecycle management across a large enterprise. We do this by understanding the prevalent forces in the enterprise and finding positive solutions or patterns that are able to balance these forces. In this document, we provide a pattern language for product lifecycle management primarily based on emergent, good, and best practices within large telecom operators.

TELECOM COMPLEXITY

Telecom operators are confronting business challenges that require deep changes within their people, systems, and organizations. With device manufacturers such as Apple, Google, and Samsung increasingly providing their own communications services, telecom operators are facing declining revenues from their staple offerings of voice, data, and messaging. The emergence of highspeed mobile Internet coupled with devices that offer users greater possibilities in service selection and consumption requires telecom operators to build on their existing strengths while capitalizing on new opportunities. Products focused around voice, data, and messaging need to be transformed and merged with new services that revolve around the connected work and life of users. The omnipresence of Internet Protocol (IP)–based networks makes it easier to create high volumes of diverse multidevice products, which will increase the complexity in the conception, implementation, operation, and retirement of products. Current inefficiencies will be further amplified given the product volume and diversity, and new practices will need to emerge in order to manage this unprecedented scale of complexity. Each of these new practices will have to be applied with other practices, creating an interconnected network of practices. These practices can help telecom operators effectively transform from value chains to value networks. Telecom operators have traditionally tried to manage product lifecycle processes through individual and independent projects, using waterfall or agile management methods. Products realized in environments that use a waterfall methodology have distinct phases for product innovation, product feasibility assessment, product implementation, product operation, and product retirement. Matrix organizations are often constructed to complement these distinct phases, and particular responsibilities are assigned to each phase. Distinct organizational units within the matrix organization have clearly defined roles and responsibilities, each producing a specific aspect of the final product. A single department that translates all business needs into designs (or a single group that manages all the data in the enterprise) is an example of these distinct organizational units. Such structures are useful for producing products that have similar characteristics.

Historically, most commercial product offerings have had similar structures, making it possible to create relatively repeatable processes for their lifecycles. The focus of such organizational structures tends to be around product delivery rather than product retirement. These processes are analogous to the classic Ford Model T assembly line, in which each person in the assembly line had a defined role, received fixed inputs, and produced fixed outputs to achieve consistency. Figure 1 illustrates the treatment of product lifecycle processes in a waterfall method.

Figure 1 — Phases in the lifecycle of a product within a waterfall methodology.

Products realized in agile environments follow more fluid and flexible phases, where each phase has a tendency to overlap with other phases. There is recognition that features that seem very important for a product at an early stage might seem less important at a subsequent stage. Through understanding the complexity of tasks to realize a product, development can be prioritized by tackling the most complex aspects of a product first. Documentation and service-level agreements are kept down to a minimum and replaced by incorporating suppliers and partners as extended members of the production system. Processes themselves are constantly being adapted in the face of changing situations by practitioners who continuously apply improvements in their methods of working, following principles like "5 whys is 1 how." However, agile work environments are difficult to achieve in highly distributed teams within complex organizations. The inspiration for agile methodologies can be traced to the Toyota Production System as realized in the late 1970s, which introduced the Kaizen philosophy of production. Figure 2 illustrates the treatment of product lifecycle processes using agile methods.

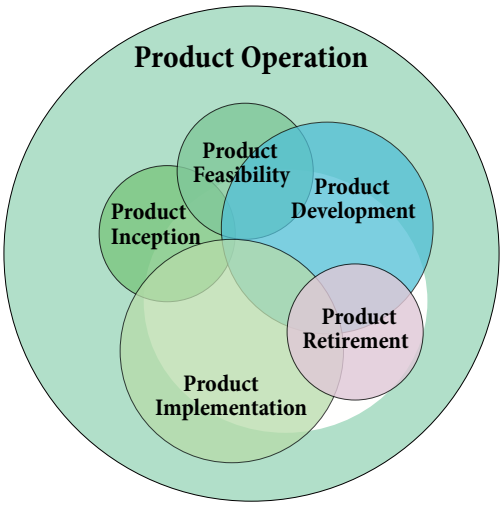

Figure 2 — Phases in the lifecycle of a product within an agile methodology.

Both waterfall and agile methods provide practices for structuring organizations for efficient product lifecycle management. There are other practices that can be applied to managing people, technological choices, organizational structures, organizational culture, and physical workspaces. These practices can be selectively applied in different combinations based on different situations and organizational cultures. The practices themselves can be classified based on their maturity. Best practices are typically established in situations where the correlation between cause and effect is obvious to all. Good practices require some analysis to establish a stable relationship between cause and effect. Finally, emergent practices typically come about when retrospective analysis is done on the correlation between cause and effect.

A pattern language can be used to describe such practices, indicating best practices with three stars and good practices with two stars. (If we had featured any emerging practices, we would have assigned these one star.) We first give a synopsis of the scale of complexity telecom operators confront today in each phase of a product's lifecycle. Next, we provide two examples from our pattern language, outlining the problem the pattern solves, the pattern's solution, and the linkage of this individual pattern with all other patterns. By using a pattern language, these practices can be woven in myriad ways for different products within the same organization, and the unique challenges a particular telecom

operator faces can be resolved using a unique implementation of these patterns.

DERIVING THE PATTERN LANGUAGE

The primary goal of developing a pattern language for telecom operators is to provide a flexible method for better dealing with the scale of complexity they confront in using new practices. The adoption of these new practices can require new organizational structures, changes in employee roles, the creation of new employee roles, and the gradual removal of existing practices. All these requirements for the adoption of new practices can be very difficult to achieve if the benefits of the change are not clear at every level of an organization. Patterns of change must be able to incorporate existing practices by absorbing their strengths and minimizing their weaknesses. Each pattern must be able to articulate the existing problems based on real situations individuals encounter, the solution for these problems, and the linkages of the pattern with other patterns.

In order to derive this pattern language, we have applied the Embodied Making method. The process of design with embodied making is intitiated by having conversations with people in the space where we want our designs to live. It starts by capturing stories, which are anecdotes as people relate them to us, and then faithfully recording them. We conducted around 120 interview sessions with people performing different roles in these organizations, ranging from customer service agents to product developers. All these individuals were involved in some capacity in the realization, operation, or retirement of products. From these interviews, we were then able to derive the prevalent forces from the processes of product lifecycle management. For instance, the following are some of the forces evident during the inception of a product:

Reluctance to discuss immature ideas
Fear that ideas will be underappreciated
Concern that competitors may copy ideas
Desire to obtain idea feedback from experts
Inability to identify experts
Tendency of experts to prefer giving direct and personal feedback
Tendency to forget about ideas after a while
Skepticism of unconventional or disruptive ideas
Tendency of commercially successful products to get more attention

We have thus far identified around 800 individual forces that are common in most Telecommunciations Operators. These forces did not manifest themselves in isolation but rather in combination with many of the other forces. For instance, one reason for the reluctance to discuss immature ideas is the fear that the ideas will be underappreciated. A solution for resolving these forces would be to create an environment where anyone can submit an idea anonymously. However, this solution conflicts with other forces, such as the desire to get credit for good ideas and the fear that others will get credit for ideas that are not their own.

Another solution that resolves these forces is to create a forum where individuals can choose to submit their ideas either openly or anonymously, with the option to reveal their identity at a later stage. However, anonymity doesn't allow individuals to receive direct and personal feedback from experts, and it doesn't resolve another force — the tendency for a single individual to have limited insight about all aspects of a product. A solution that does resolve these additional forces is to require that for any idea to be adopted, it must be championed by three individuals who collectively consider its commercial, operational, and technological aspects. This continuous process of assessing solutions eventually provides a set of stable solutions that work well together, and each of these solutions is then restructured into a pattern. Wherever possible, the patterns are named after a common theme, such as a garden or town. As new patterns emerge, existing patterns are reassessed and redesigned to reflect their individual and collective interaction with other patterns. Together the patterns provide an interwoven pattern language.

01 Aware Instances

Components are aware of their role in fulfilling products, and product instances contain their own unique semantics. Awareness isn't elsewhere, but with the component itself.

A product instance, or a product as a customer experiences it, is realized through several systems, ranging from configuring access in a network switch to a customer record in a customer-relationship management system. Each of these systems have a part in constructing the customer's experience of the product. Network equipment in telecommunications environments historically have a lot of variation, often supporting standards from the 1980s such as GSM to contemporary standards such as LTE. With the move to an all Internet Protocol (IP) network, where will be an even greater diversity and volume of products with providers moving beyond voice and data to content (music, movies, books, education, etc.), several configurations will be possible in each of these system. This will lead to high-levels of complexity in managing all of these combinations. Processes for provisioning products need to anticipate several eventualities, to the extent that process definitions need to be conscious of the state of several network elements and systems simultaneously. When controllers are used to control the a number of states, the controllers need equal or greater knowledge than the total number of states. Similarly when product, customer, and resource specifications are used to define the behaviour of instances, the specifications need knowledge equal or greater than the sum of instance variations. When provisioning processes are treated as end-to-end transactions, errors in the process necessitate restorations of original system configurations before the product was provisioned through an expensive rollback. Rule-bases for individual situations such product pricing (promotions, discounts, fees, allowances, commissions, etc.) or permitted products given an individual's installed product base further increase complexity. In the highly distributed environment of telecommunications organizations, several components are able to change each other's states, and controllers and specifications need to consider meaningful combinations of state. However if the controllers and specifications are not aware of a combination of state, which can number in the several thousands, systems recovery can only be established through restarts. Given that stable systems require their control mechanism to address as many variations of state as possible, creating centralized controllers and single points of truth only increase the complexity, not reduce it. "Things fall apart, the centre cannot hold."

Ross Asby, the creator of the law of requisite variety, succintly summarized "variety can destroy variety". Ashby, W.R. 1958, Requisite Variety and its implications for the control of complex systems, Cybernetica (Namur) Vo1 1, No 2, 1958.

For a thorough analysis of distributed state control, see Moseley, B., Marks, P., Out of the Tar Pit, 2006.

From the opening lines of "The Second Coming by W.B. Yeats: Turning and turning in the widening gyre, The falcon cannot hear the falconer;
Things fall apart; the centre cannot hold;

Therefore make product instances exist integrally where they have knowledge of the components that realize them, and conversely, the components that realize them have knowledge of the product instances that reside within them. Instances are aware when they are unfulfilled, and are capable of being fixed individually. Make it possible for instances to have individual and unique behaviour, and for customers to have their unique situations reflected in the system. When systems aren't capable of enabling unique behaviour for instances, utilize proxies for in systems with that capability.

Use the Language for Product Instances to reflect product instances and the customer's situation and use the same Language for Product Instances to interpret the customer's unique situation and install base. If the customer is experiencing problems with their product, permit customer agents to Change it with Confidence within the Fortified Town. Enable Product Markers that describe the product instance's performance, costs, and efficiency.