LONG-TERM MANAGEMENT OF CONFIGURABLE TELECOMMUNICATIONS SERVICE OFFERINGS: A CASE

Author(s): Juha, Tiihonen

Abstract

Long-term management of configurators has been identified as challenging and business-critical in previous work. Configuration knowledge and prices may change frequently due to product changes and business reasons. If sales rely on configuration support, prompt updates are important, even business-critical. This work reports challenges and practices of long-term management of a configurable B2C broadband telecommunications service offering, contributing one of the first studies on long-term management of service configuration knowledge. Some significant differences to management of physical configurable offerings were identified, especially in the area of installed base management.

To evaluate system effects, a change in a telecommunications service offering must be considered from at least three dimensions: how it affects the products, how it affects prices, and how it affects management of existing customers (the installed base).

Five processes related to offering changes were proposed and characterised. A single process cannot be optimal for all product and pricing changes, as the scope and effect of changes can vary significantly. A lightweight enough process should be used for small and frequent changes, and an appropriately rigorous process should be used for significant changes. An adequate set of processes may to some extent alleviate problems related to an unacceptably slow ability to react to market changes.

Keywords

Configurable services; configurable products; configurator; long-term management.

1 Introduction

1.1 Problem and Background

Long-term management of configurators has been identified as a challenge in previous work. Configuration knowledge often changes frequently due to product changes and for business-related reasons (Fleischanderl, Friedrich, Haselbock, Schreiner & Stumptner, 1998), and pricing changes may be frequent. If sales rely on configuration support, fast updating of configuration knowledge is important, even business-critical (e.g., Barker & O'Connor, 1989). Problems related to configurator introduction or long-term management might delay new product introductions or product improvements (Barker & O'Connor, 1989; Tiihonen, Soininen, Männistö & Sulonen, 1996). More complexity arises if reconfiguration of existing product individuals needs to be supported (Männistö, Soininen, Tiihonen & Sulonen, 1999).

This work reports challenges of long-term management of a configurable B2C broadband telecommunications service offering. Change types that are relevant to the case company and their system effects are discussed, and a set of processes is identified.

1.2 Method

A case study with eight themed interviews was performed. The informants included business and product management; persons that manage, maintain, and develop configuration models and related prices; senior consultants from the ERP system vendor; and one person from organisation responsible for maintenance of information systems. The interviews were recorded and transcribed. Analysis was performed with Atlas Ti 4.2 software to collect answers related to interview themes, and to analyse spontaneously emerged topics. Document analysis was performed on documents, describing business requirements for the configuration model and its detailed implementation. Some ERP system vendor documents were used as a reference. A set of processes were constructed to address existing challenges and identified change types.

The rest of this paper is structured as follows: Section 2 gives an overview of the case company, related information systems, and frequency of changes; Section 3 analyses different kinds of changes; Section 4 outlines problems in change management of the case company; Section 5 describes a set of processes for change management; Section 6 discusses the results; and Section 7 contains our conclusions.

2 The case company

Our anonymous European case company had hundreds of thousands of broadband customers. The case offering consisted of broadband connections with varying speeds and delivery technologies. Each broadband connection had a set of additional service elements such as e-mail and internet newsgroups bundled with the subscription. There were a few different bundles; cheaper and slower subscriptions contained the least extensive service element bundles. Furthermore, a relatively large number of selectable service elements were available. The company used a recently deployed ERP system with a configurator module.

Business and product management were responsible for the offering and pricing. The company had a maintenance organisation that was responsible for maintenance and development of operative systems. Changes that require software development were managed by the maintenance organisation with methods and processes that suit software development. Business and product management were familiar with this process, and it was established. Changes requiring development were prioritised and assigned to development projects in change management boards on the basis of change requests and feasibility studies. There were two separate boards for different business areas. They prioritised changes for systems that were partially the same, including the ERP system. This created competition of available resources, and usually an equal number of changes were selected to be implemented from both boards. Development projects were usually outsourced. As a result of a development project, a new ERP system release was produced.

There was a group that was responsible for the configurator (e.g., modelling and pricing set-ups, and configurator integrations). The group was not part of the maintenance organisation.

From the view of business management, implementing offering changes took a long time in relation to change triggering pressures in the market, and as a whole the company could not react fast enough. There was a need to be able to perform product and pricing related changes faster and more flexibly. There was a need to identify problems, sources of errors, and to improve processes for managing offering-related changes.

2.1 System architecture overview

Several information and network systems implemented the broadband services offered by the company. An ERP system was used to order the services and to manage customer information and information on existing configurations. The ERP system was adapted to the telecommunications domain by the vendor. It was configured according to local requirements, and integrated with several other systems. The configurator was a modular part of the ERP system.

Ordered services are made available to a customer (connect), and availability is removed when (e.g.) a subscription ends (disconnect). This is called *provisioning*. For consumer products, provisioning was automatic except for physical connections. Provisioning was performed by systems external to the ERP system – integrations delivered information on items to be provisioned.

Configuration models and end-user (e.g., customer service) interface definitions were modelled with a configurator modelling tool and stored in the ERP system's database. All broadband products were modelled within a single configuration model. The configurator can call external routines. Units of code specific to both the generic telecommunications domain and the case company were used. Often such units of code were related to system integrations such as provisioning.

Pricing functionality was a separate modular part of the ERP system. It determines a price to a configuration on the basis of made selections, campaigns, and other possible discounts. Billing is managed by a separate system. Both ERP and billing "know" the pricing structure and campaigns, and the systems must have a common view on prices.

In addition to production system instances, there were separate development and test instances—implying a need to transfer information between them as change implementation proceeds.

2.2 Effects of changes to systems

Creating capabilities to deliver an updated offering may affect several systems, which may require system set-ups, software development, and hardware and network changes. These efforts may be more significant than ERP, the configurator, and billing changes. Thus, it is not possible to concentrate only on the configurator when deciding how service product changes are to be managed.

Set-up changes were performed by updating data or parameters within system(s). The ERP system and configurator enabled the company to change products, available service elements, and related configuration rules by configuration modeling. In addition, the configurator end-user interface is defined with set-ups. In other words, set-up changes can significantly affect the offering and the user interfaces, due to the power of configuration and user-interface modelling. Therefore testing plays an important role, even with set-up changes. It was seen that changing the user interface may take a long time compared to other set-ups. Furthermore, pricing and campaigns were changed with set-ups. Pricing logic had to be deployed synchronously in the billing system and the ERP system. In the ERP system, price modifications were effective immediately. The billing system provided a possibility to create price lists in development mode and to publish them when needed. The configurator enabled publishing product setup-changes "on—the—fly." However, publishing changes had been performed during downtime.

Some changes required rebuilding the ERP system. Deploying new software code usually required a system build, implying downtime. However, it was possible to deploy new or updated code called by the configurator without a rebuild.

It was seen that some changes need regression testing, while others need none, or a minimal amount. Regression testing was a manual resource-intensive process. It was considered that changes that do not require significant regression testing could be implemented independently of system releases to enable a faster schedule. Automation of regression testing was considered a potential way to speed up changes, and was being planned.

Major changes included developing significant new functionality, significant coding, provision logic changes, or interface-related development. Testing extended to unit, system, integration, and user-acceptance tests. These changes were implemented as projects that produced a new ERP system release. Some desired functionality enhancements would have required customisation of the ERP system, which was strongly avoided.

It was expected that some configuration model changes would require updates to the installed base to remain consistent with the new configuration model. For example, items would have to be added to or removed from configurations. The number of affected customers would often be significant, requiring programmatic updates called *mass-updates*.

2.3 Frequency of changes

The frequency of changes depended on market situation. Growing markets tended to generate frequent changes due to competitive actions. According to rough estimates, at least 70% of changes were pricing changes. New price lists were introduced monthly. New product features occurred roughly once in every two months. Additionally, a new ISP product was to extend the broadband configuration model significantly. Services or service elements were terminated relatively seldom: one service element had been terminated during the last 1-2 years.

Three or four maintenance releases of the ERP system had been agreed upon for 2006. In each release, 5 change requests per change management board were to be implemented. The maintenance organisation had 136 open change requests related to the whole ERP system. About 30 were considered "urgent," 55 "very high," a few "low," and the rest "high." The number of configurator-related change requests was not specified.

The broadband configuration model had been updated roughly 1-1½ times per 2-week period since the production use of the ERP had begun (interviews took place within 6 months of production launch.)

We were unable to reliably estimate the total effort related to management of changes.

3 Changes

A telecommunications service product offering change has at least three dimensions: how it affects the products, how it affects prices, and how it affects management of existing customers (the installed base). A change is often related to several dimensions.

3.1 Service product changes

We identified several types of product changes (the list is not exhaustive):

- New product generation.
- New product.
- New separately available service element.
- New bundled service element (plus removal of a bundled service element).
- Substitution of a service element or product.
- New bundle.
- Change of a service element.
- Discontinuation of a product.
- Discontinuation of a service element.

Introducing a new product generation implies significant differences to existing products. It is necessary to consider issues such as whether the new generation is modelled in the same configuration model, if sales of the previous generation will continue, and if migration of existing customers is needed.

Introduction of new product(s) is always a significant change in the offering. However, some new products may be structurally very close to old ones. For example, a new top speed connection could be introduced and modelled almost identically to a previous “high-end” product.

At least two ways to introduce a new service element exist: adding the new service element to bundles of affected products, and making the new service element available separately (e.g., as an optional item in certain products). The service could also be bundled in higher-end products and separately subscribed in other products, which would create additional complexity.

Service elements that are part of a bundle are not managed individually in the configurator or in the ERP-installed base. Component services of a bundle are shown in the configurator user interface for information purposes, but they are not directly selectable. Provisioning systems provision the bundle as a whole. The ERP system’s installed base “sees” only the bundle type. When bundle package contents are changed, it is enough (with regard to the configurator) to update only the configurator attributes and related configuration rules that indicate which service element belongs to which bundle. In addition it is necessary to modify provisioning systems to treat the bundle appropriately. However, a mass-update in provisioning systems would be required to provision the new bundle contents for current customers.

The desired availability of the new value-added service in existing products affects how the service will be introduced to the systems:

- If the new service element becomes mandatory in some products, it would be easiest to manage through bundles. Adding the new service as a separately orderable but mandatory service element would require extra effort compared to the bundle approach.
- If the new service element becomes a non-substitute, separately orderable service, it will be added to the configuration model as a new item. Corresponding provisioning and pricing items must be created. A mass-update to the installed base is probably not needed, because previous installations do not exist, and the service is not automatically added to existing customers. Adding the new service element that would be by default included in subscriptions is probably uncommon.
- Management of the substitution of an existing separately orderable service element with another could imply different scenarios. If the substitution is "genuine" both for the telecommunications company and for customers, it could be possible to manage it by renaming service elements in the configurator. Provisioning and a mass-update of existing provisioning would have to be performed. However, if substitution requires customer consent, it would become necessary to offer customers a choice, and a combination of introducing a new service element and removing an old one would be required. Customers who accept the substitution would be mass-updated to the new service, and others would be disconnected.

If the new service element is configurable, relevant configurable structures or attributes must be defined, and changes to the user interface are more significant. Further, provisioning logic and pricing must take these aspects into account.

Necessary constraints to represent business rules must be created. If the new service element is alternative to some other service elements constraints must be created, because the configurator does not support the concept of alternative elements.

Adding a new bundle has significant effects in the configuration model, and provisioning systems would have to be modified to manage the new bundle type. It is also possible that the existing customer base would have to be mass-updated, to have the new bundle provisioned to intended existing customers.

Change of a service element relates to modification of some detail(s) of a service element, for example increasing the size of the e-mail inbox. With regard to the configurator, some of these changes are not visible, and some may introduce (typically) local modifications.

Discontinuation of product(s) or service element(s) sometimes takes place. There is often an attempt to avoid these changes: customer relationships may be negatively affected, affected customers must be informed, documents given to customers such as service descriptions may have to be updated, and sometimes even the contract must be updated. Customers should be offered substitute products, and transition to new products must be managed. This increases the complexity, risks, and efforts required. Discontinuations were seen to become more common due to increased requirements of cost-efficiency –some service elements create significant costs. Termination of even a minor service element concerns thousands or tens of thousands customers. In principle, terminating services would cause reverse changes in systems compared with introducing new ones. In practice, it may be desirable to leave the existing elements in place and mark them obsolete, hide them, or otherwise arrange them so that they will not be used. Business rules may have to be defined in the configurator to ensure that correct values are given to some obsolete configuration parameters. The installed base must be managed to remove provisioning.

3.2 Pricing changes

Price-list changes affect a set of individual price(s) on price-list(s). These are set-up changes in ERP and billing. New price items are added to price lists when new products or charged additional services are introduced. These are set-up changes in ERP and billing systems. Systems that collect billing data may be affected.

Campaigns and special offers are typically temporary price reductions on initiation fees, additional services, or product prices. In campaigns, initiation fees and monthly prices of products and/or additional value services are reduced for a fixed period. Equipment or other benefits may be offered at discounted prices or for free. For example, half-price for the monthly charges for the first three months, and/or free equipment may be offered. Benefits may be subject to various conditions that must be met. Campaigns are managed with the ERP system and the billing system. Defining campaign conditions, and the need to avoid overlapping campaigns, cause some additional complexity.

A free service element may become a charged one. Such a service element has probably been part of one or more bundles. If periodical (e.g., monthly) charges were created, the service would probably become optionally available by a separate subscription, and managed accordingly. Business rules for preventing ordering of charged services from additional user accounts could be affected. If only pay-per-use components were added, they would probably be managed outside the configurator. The service could be kept in bundles if continued availability of a charged service is possible. If not, it should be made separately available by subscription, and a mass-update would be needed.

A charged service element may become free. Charged services with periodic charges would probably have been available as separate items. Such a change could be managed easiest by setting prices to zero. Another option would be to move the service element to relevant bundles, but this would incur extra work, and the installed base would have to be mass-updated to remove subscriptions to the service moved to a bundle. Provisioning logic and billing would be affected.

3.3 Management of existing customers

Change management of existing customers may involve communications, management of changes at the level of individual customers, and mass-updates to the installed base.

Customers must be informed of significant changes well in advance; e.g., terminating a service element is to be notified half a year in advance. Sometimes the desire to make changes was held back by the need of customer-specific communication.

The following types of changes with respect to management of individual customers were identified. Each type may have a different schedule, customer service or technical support workload, and system implications:

- No required customer communication.
- General communication that is not targeted to specific customers. For example, news on available new service elements can be included in regular customer communications or marketing materials such as newsletters.
- Communication targeted to selected customers, or different communication to different customer groups. For example, informing specific affected customers is required.

- Need to offer a decision point to each customer. For example, each customer may have to be offered alternative ways of reacting to a price or product change. This requires managing the customers individually, which significantly affects the amount of work.
- Modifying or renewing the customer contract. A contract change always requires contacting the customer, and involves risks as the customer may reconsider continuation of the customership.
- Need to change customer equipment or environment; e.g., a new modem could be required to facilitate some changes to specific customers. This implies individual management and often the need to offer a decision point.

3.4 Mass-updates

Mass-updates are performed by manipulating configurations of the affected customers with external programs that access the configurator through an application program interface (API), and that cause provisioning of the resulting changes. Great care must be exercised to maintain all configurations consistent. It is necessary for customer service to know the status of each customer with respect to the mass-update. Mass-update programs must communicate their state in a rigorous manner so that customer service can know if the change has been applied to a specific customer, when it will happen, and how it will affect the customer.

It is critical to mass-update the correct set of customer installations. However, the most typical error in mass-updates was selecting a wrong set of customers to be updated. According to one informant, most of the time related to an offering change may be consumed by updates of individual installations.

Members of a team responsible for configurator maintenance considered mass updates to be potential sources of significant problems. Errors are easily created, and a high load may be caused to support organisations. Mass-updates may cause performance problems or even prevent making new orders when a system becomes bottlenecked. Customer groups with some type of exception that requires specific handling were considered to be especially prone to errors.

4 Identified problems

4.1 Processes and organisation

The process for performing changes, from an initial business need to deployment, was not fully defined. Separate process descriptions or operational guidelines existed in different parts of the organisation, but they were not comprehensive, and varied in level of detail. The total change process was, as described by one informant, “derived from the organisational structure”. There was uncertainty about existing alternative processes, and roles of different stakeholders in each process. Relatively few persons were aware of how to initiate a change request process, and procedures often had to be explained personally. Better coordination of changes, from identified business need to a deployed solution, was desired. It seemed that potential for using lighter processes for minor changes was not utilised in a properly defined way.

Highest priority was assigned to too many change requests, and there were too many justified change requests in relation to available resources. Development resources limited the number and scope of changes that could be implemented. Furthermore, solving significant performance problems had taken priority over other improvements in the maintenance organisation.

From maintenance organisation’s point of view, change decisions were made in multiple locations “all over the house.” Persons responsible for pricing set-ups had a similar view – too many persons were empowered to ask for changes, and no unit coordinated the requests. Change requests that tried to pass the normal process had been given to implementing stakeholders. Ad hoc meetings were arranged to further some changes. This was considered a serious problem.

New persons in business management were not aware of possibilities and limitations of the systems in use. Sometimes system limitations created a need to alter the desired change specification. These issues caused additional iteration in the change processes.

Most configuration model updates were bug-fixes. Bugs had been introduced while manually entering configuration model data to the production system from an Excel-based master document. Errors included mistyping, or forgetting to enter an item, rule, or attribute value. Automatic transfer, from test or development environment to production, was not yet deployed.

The ERP system and configurator maintenance operations were organised in such a way that rapid testing and deployment was not genuinely in the interest of all stakeholders. Stakeholders had different views on appropriate balance between flexibility and predictability of testing schedules. It was felt that friction between organisational units and lack of flexibility reduced the company’s capability for reacting quickly to events in markets. Overlapping tests were performed by different organisational units, which created delays. On the other hand, campaign setups were not tested, possibly due to personnel background. Errors in campaign set-ups were considered to be common.

4.2 Change documentation and tool support for changes

The quality of change requests varied. Often meetings had to be arranged or several phone calls made to analyse the real business requirement. Sometimes requirements were incomplete or unclear, and easily misunderstood. Logical inconsistencies in change requests were considered common. It was felt that sometimes new ideas were added light-heartedly to a previous version of a change request, making it inconsistent. Due to these reasons, after a change had been implemented and tested, it had sometimes been noticed that the implemented change is not what was actually desired. Sometimes ambitious persons used lots of effort to implement “great” features that were not needed. This was attributed to change requests leaving room for interpretation.

Stakeholders such as product management, maintenance organisation, and configurator maintenance used their own set of tools for change management, if any. Thus, there was no clear repository for change request and product information, and no IT support for workflow management. Information flows were not always adequate. For example, changes had been decided or implemented based on old information.

A significant challenge for configuration model documentation and specification existed: a master view of configuration models had to be maintained outside the configurator in spreadsheets. The spreadsheets were based on the ERP vendor’s recommended templates. They were to contain the same information as the configurator, augmented with implementation-status colour coding. Entering configuration model changes manually from the spreadsheets was a significant source of errors and included repetitive work. The spreadsheets were considered hard to understand, and cumbersome to use and maintain. Reports from the configurator were difficult to compare to the spreadsheets. Similar but less significant challenges were related to pricing master spreadsheets. Configurator user-interface definitions were not documented, which was considered a problem by several informants. An efficient documentation method was called for.

The test environment did not fully correspond to the production environment. Thus, it was not possible to detect some errors in testing. Interfaces across systems had caused several errors that had been detected only in production phase. For example, a returned value was different from what was expected, or some logical error had been made in design. Often such errors concerned special cases. Often outdated interface documentation contributed to these problems.

Self service channels to customers were separate applications with a web-user interface. They were considered prone to errors and hard to test, and prices were hard-coded, which caused additional errors. It was considered that the configurator sometimes behaved differently when calling it programmatically (e.g., from self-service channels) instead of the normal user interface.

4.3 Consequences of errors in changes

Errors that are small from a system perspective often caused a need to manually adjust customer installations or manual intervention to order progress. It was considered hard to repair errors in a production environment. Errors sometimes caused a high load on customer support, which in turn could cause long waiting times and poor customer service experience. Such a situation had been "very painfully visible to customers".

Sometimes it was possible to fix errors by developing software that fixed the consequences of an error. A significant amount of work could be consumed trying to prevent problem visibility to customers. If, for example, service availability is lost, economical compensations may be required.

Order entry had been prevented due to erroneous set-ups. For example, the system does not accept negative prices. Therefore accidentally simultaneous campaigns that produced a negative price prevented order intake. When ordering is not possible through the self-service channel, the order may be permanently lost. Customer-service can work for a limited time by taking relevant information to paper, and entering data afterwards. However, extra work and delays are caused. Erroneous or incomprehensible bills had been sent to customers due to campaign setups.

5 Proposed change management processes

A single process cannot be optimal to all product and pricing changes, as the scope and effect of changes vary significantly. A lightweight enough process should be used for small and frequent changes, and an appropriately rigorous process should be used for significant changes.

The traditional waterfall-process model seems appropriate with regard to ERP and the configurator, billing, and integrations, because product changes, once appropriately and feasibly defined, are relatively stable, and because system changes caused by most offering changes are relatively straightforward.

Five processes related to offering changes were proposed and characterised: routine price-list changes, routine campaign setups, set-up changes, minor development, and significant development. When changes are not limited to routine pricing changes, a change request (*CR*) is created to initiate the change process and to document the desired change. A CR should specify the business case and the goals to improve communication on what is actually needed, and to partially facilitate testing against business requirements. Based on a CR, a feasibility study is performed to analyse the change: e.g., is it possible?; what is the probable solution, or what alternative solutions exist?; what is the estimated effort?

For routine pricing and campaign set-ups, the process selection is obvious. Deciding whether set-up, minor development, or significant development is needed is based on the nature of the change and on efficiency (discussed in Section 6). Product management with an implementation team manager makes the decision between these processes. Changes with high-enough priority are assigned to the implementation processes. Design, actual implementation, testing, and documentation are performed appropriately for each process. Each process is characterised in the following subsections (Table 1 gives an overview of some process characteristics):

Process / Step	**Price-list changes**	**Campaign set-up**	**Set-up changes**	**Minor development**	**Significant development**
Initiation	Pricing decision, no CR	Campaign decision, simplified CR	CR	CR	CR
Feasibility study	No	No	Yes, limited	Yes	Yes
Prioritisa-tion	No CM board. Implementation team level	No CM board. Implementation team level	No CM board. Product management & implementation team manager	No CM board. Product management & implementation team manager	Change management board
Assignment to implemen-tation	Business management directly to implementing team	Business management directly to implementing team	Product management agrees to schedule with the manager who is responsible for configurator set-ups & small changes	Product management agrees to schedule with the manager who is responsible for configurator set-ups & small changes	Changes packaged to development projects. Possibly tender-based outsourcing.
Design	None	Typically none, routine design for complex campaigns	Routine design, possibly set-up prototyping to find the solution.	Routine design, possibly limited prototyping	"Full" design required in implementation projects
Implemen-tation environ-ment	Production	Production, or complex ones in test environment	Development or test. Transfer to production	Development or sandbox, then test. Transfer to production, rebuild possible	Development, then test. Produces system release
Testing	Verify with a number of test orders. Tools that compare or enter prices in ERP and billing systems.	Test affected and typical orders. Complex campaigns with a peer-review and test environment	Test typical orders and orders affected by the change. Full regression testing for complex changes	Regression and unit testing, integration testing if interfaces changed. Set-ups tested like in set-up changes	Full including regression, unit, system, integration, & user-acceptance testing
Documen-tation	Written price decision, price list master worksheet	Decision on campaign features in written form. The simplified change request documents the campaign	Change request and feasibility study. Updated configurator and pricing set-up documents. Record of performed testing	Design docu-ments, CR and a feasibility study. Updated configu-rator and pricing set-up documents. Record of performed testing	CR and a feasibility study, full design and implementation documentation. Record of performed testing

Table 1: Characteristics of proposed change processes

5.1 Routine price-related changes

Price list changes and campaigns were frequent and covered roughly 70% of changes. Thus processes that make these changes fluent are required. Typically once every month some prices are updated, making it possible to predict forthcoming price list changes and to allocate resources in advance with accuracy. In the price-list-change process, only prices of existing price elements are changed.

Campaigns are frequent, often urgent, and occur unpredictably due to competition. The logic of campaign set-ups requires basic data input, or sometimes more advanced set-ups that include building logical conditions that define the campaign. Only campaign set-ups that are supported by ERP and billing system set-ups are allowed in this process to maintain the straightforward set-up. A limited change-request document is created to verify and to document the campaign.

In terms of system effects, price list changes are very simple, and campaigns are usually simple set-up changes. It is necessary to update set-ups both in ERP and in the billing system. Therefore it is important to synchronise the changes and to avoid inconsistencies, and it was recommended to develop tools that enable entering the prices only once. Mass-updates are not needed, and management of existing customers does not need individual responses or contractual changes.

5.2 Set-up changes

In this process, local configuration model changes are performed, or pricing setups are changed in a more complex way than allowed by previously introduced processes. Product details may also be set up in other systems. No software development is allowed. Management of existing customers is not allowed to require individual responses or contractual changes. The configurator end-user interface may not be changed in such a way that user training would be required, because that may be better managed in conjunction with a new system release. Mass-updates are not allowed, because developing mass-update routines would be required, and because mass updates are error-prone, major operations. Similarly, interfaces and code called from the configurator are not allowed to require changes. As a result, changes that require rebuilding systems are not performed. Examples of setup-changes are: changing a default value in the configurator, increasing e-mail box sizes, and pricing changes that introduce new price items.

A change request and a limited feasibility study to verify the set-up nature are required. No change management board prioritisation is required. However, there may be a need to prioritise the set-up implementation in relation to other activities. Schedules of more significant changes may be affected; therefore it is a business decision to use this process. Design is limited to routine design activity that may involve set-up prototyping to find the optimal solution that works as expected. Configurator and pricing set-up documents are updated. A peer-review of design is recommended for at least the more complex set-up changes. Appropriate testing of changes in the test environment must be possible. For simplest set-ups limited regression testing is performed, to validate that typical orders, and orders that are affected by the change, work as expected. For complex set-ups, comprehensive regression testing is performed. A record of performed regression testing is kept.

Because changes are performed with set-ups, there is little if any risk of affecting the technical contents of ongoing development projects, which makes the changes relatively easy to manage. Introduction of new service elements is seldom possible, because capability to produce the service usually depends on development in other systems, or provisioning would be affected beyond set-ups. Similarly, changes of bundle contents are not possible, because mass-updates would be required to provision the changes. Termination of subscribeable services may be better managed in other processes, because customer contacts with long lead times are required. Automated regression testing would facilitate more frequent use of this process.

5.3 Minor development

Between pure set-up changes and significant "full-scale" changes, there is a class of product and pricing changes that can be implemented with small and straightforward software development activities and set-up changes. This process has similarities with both set-up changes, on one hand, and with significant development, on the other. The estimated total amount of work should be small compared to projects that produce a release. The nature of programming should be restricted to local or otherwise straightforward issues that are easy to test. It is required that adequate regression testing in the test environment is possible, and changes that would require full-scale integration testing are not allowed. The implementing persons must be familiar with related existing code and the system environment. A view was taken that minor mass-updates can be allowed in this process. This requires that there is high-enough confidence that the resulting mass-update can be performed reliably and without significant performance problems. Management of existing customers is not allowed to require individual responses or contractual changes. Thus there are no significant issues that affect the load of the customer service.

A change request initiates this process, and a feasibility study is performed to identify a solution and to verify that the change belongs to this process. Product management, along with an approving manager responsible for configurator set-ups, makes the process decision based on the feasibility study and business urgency. This ensures that the change is small and easy enough to be performed in this process, and that there are business-based reasons for using this process. Due to limited change scope, prioritisation bypasses formal change management board processing that introduces major delays. Careful management is required to solve resource conflicts with longer-term development and to establish a viable schedule. Routine design is needed to specify the software and set-up changes. Changes are implemented in the development or "sandbox" environment. It is recommended to use peer-review for software code and set-ups. Regression testing and unit testing are needed. If integrations are changed, they must be tested. Set-ups are tested and documented similarly as in the set-up change process. Documentation includes the change request and the feasibility study, design documents, updated configurator and pricing set-up documents, and a record of performed testing.

If automated regression testing could be implemented it would be easier to utilise this process, because the amount of overhead in implementing changes would be lower. A system build is usually implied, which in turn implies some downtime when changes are taken into production.

5.4 Significant development

This process is the full-scale change process. It is to be used if any of the following conditions are met: significant software development is required, interfaces are significantly modified or new interfaces are developed, full-scale mass-updates are needed, elevated customer service or technical support load is expected, or user training is required. A significant amount of resources can be consumed in development. All changes that product management and business management have not assigned to other processes belong to this process.

A change request is created to specify the change, and a feasibility study is required.

Development prioritisation in the relevant change management board is required to allocate development resources to most significant change requests. As the result of prioritisation, selected changes are packaged to a development project, or, in case of a very significant change, a separate project may be initiated. Projects may be outsourced through a tendering process. Actual design is performed within implementation projects. Implementation is performed in development environment(s). Full, appropriate testing, including regression, unit testing, system, and integration tests, as well as user-acceptance testing, is required. The test cases should cover "white-box" test cases, defined by technical experts, and "black-box" business-originated test cases. Product management should participate in defining relevant business-derived test cases. Full design and implementation documentation is to be produced.

6 Discussion and comparison to previous work

The reliability of results may be negatively affected due to limited access to actual operations beyond interviews and to documentation. Some planned interviews were cancelled, and some materials (e.g., concrete examples of change requests and feasibility studies) were not received.

The types of changes presented are by no means comprehensive. For example, different types of changes would be implied if, for example, mobile and broadband products would be bundled together, while still managing mobile products in legacy systems.

The number and nature of recommendable processes is somewhat subjective. There is potential to combine routine price-related processes, although different implementation resources are used, and testing requirements may vary. Similarly, it can be argued that set-up changes and minor development are close enough in nature to be one process. However, we argue that a set of processes is needed, and the processes must be clearly defined, communicated, and used. It must be possible to perform at least some set-up changes related to minor product element changes, with a more agile process than those developed for software development. Price and campaign set-ups are frequent and need one or two processes that are separate from other processes.

Care must be taken when deciding which process to use. If too lightweight a process is chosen, the possibility of errors increases. The lightest possible process is not always optimal in case of non-urgent changes: for example, some "minor development" changes could be more economically performed in conjunction with changes that are bundled to a release, because testing effort may be split over several changes, and less disruption may be created to ongoing implementation projects. Business units should holistically prioritise the required changes. Business benefits of the related change must be considered in relation to possible effects to ongoing development projects, and other factors affecting available resources.

In many respects, challenges of long-term management of telecommunications services seem similar to those identified in earlier work concerning physical products: configuration knowledge changes relatively frequently due to product changes, and for business-related reasons, in a similar vein as identified by Fleischanderl et al. (1998). Pricing changes were more frequent than service product changes. When sales rely on configuration support, fast updating of configuration knowledge is important, even business-critical (e.g., Barker & O'Connor, 1989). Problems related to configurator introduction or long-term management might delay new product introductions or product improvements (Barker & O'Connor, 1989; Tiihonen et al., 1996).

Configurations of physical products are typically not affected when a company changes its product families, and product individuals evolve independently of the product family as they are modified in after-sales. Consequently, the modified product individual consists of old and new components (Männistö, 2000, p. 9). Contrary to physical products, existing service configurations can be directly affected when a telecommunications company changes its product families. In other words, it is possible to update a service description and contents without physically (and often expensively) modifying an existing product individual. Second, telecommunications subscriptions cannot evolve independently of their descriptions in the service provider's database (except for physical components such as terminal equipment). In our view, these are significant differences compared to physical products.

In configuration of physical products, more complexity arises if reconfiguration of existing product individuals needs to be supported, and significant effort may be needed to enable systematic reconfiguration (Männistö et al., 1999). In contrast, telecommunications-services reconfiguration is routine in everyday business. Customers can subscribe to new service elements, change their subscription speed, etc. In our view, reconfiguration of telecommunications services is easier and more frequent than that of physical products.

In this paper, most case-company-specific suggestions were not discussed. Many of the most significant company-specific challenges were already being addressed. This paper might give an overly negative view on the situation on the company, because we largely focused on problems and challenges.

7 Summary and conclusions

This work contributed a case in long-term management of a configurable telecommunications service offering. To evaluate system effects, a change to a telecommunications service offering must be considered at least from three dimensions: how it affects the products, how it affects prices, and how it affects management of existing customers (the installed base). Furthermore, it is necessary to take other systems into account when deciding how a change is to be managed, because the capability to provision and deliver ordered services must be created or maintained.

Five processes related to offering changes were proposed and characterised. A single process cannot be optimal for all product and pricing changes, as the scope and effect of changes can vary significantly. A lightweight enough process should be used for small and frequent changes, and an appropriately rigorous process should be used for significant changes. An adequate set of processes may to some extent alleviate problems related to an unacceptably slow ability to react to market changes.

Some differences to management of physical configurable offerings were identified, especially in the area of installed base management.

Even as a customer of a major ERP and configurator vendor, and vendor's competent consultants, the company had no adequate way to represent configuration models outside the configurator, and the configurator could not provide helpful views or reports to configuration knowledge for the purposes of configuration knowledge management.

A number of significant problems related to long-term management of the configurator were identified. Despite these, a main configurator business promise seemed to be fulfilled: the configurator created consistent and complete configurations. A limited number of configuration modelling mistakes had been visible to customers. However, errors caused significant manual work and increased load in customer service. Many of the challenges and benefits of configurators identified in previous work (Heiskala, Tiihonen, Paloheimo & Soininen, 2007) seemed to apply also when applying configurator technology in a service context. Some service specific phenomena were identified.

Acknowledgements

We gratefully acknowledge the financial support from Technology Development Centre of Finland (Tekes). We thank our case company for co-operation, access to information, and financial support.

References

Barker, V.E., & O'Connor, D.E. (1989). Expert systems for configuration at Digital: XCON and beyond. *Communications of the ACM, 32(3)*, 298-318.

Fleischanderl, G., Friedrich, G., Haselbock, A., Schreiner, H., & Stumptner, M. (1998, July-Aug.). Configuring large systems using generative constraint satisfaction. *IEEE Intelligent Systems, 13(4)*, 59-68.

Heiskala, M., Tiihonen, J., Paloheimo, K.-S., & Soininen, T. (in press). Mass customization with configurable products and configurators: A review of benefits and challenges. In T. Blecker, & G. Friedrich (Eds.), *Mass customization information systems in business* (pp. xxx-xxx, chapter 1). Hershey, PA: IGI Publishing.

Männistö, T. A conceptual modelling approach to product families and their evolution. (2000). Ph.D thesis, Helsinki University of Technology, Department of Computer Science and Engineering. *Acta Polytechnica Scandinavica, Mathematics and Computing Series, 106*, Espoo 2000, p .9.

Männistö, T., Soininen, T., Tiihonen, J., & Sulonen, R. Framework and conceptual model for reconfiguration (1999). In *Configuration Papers from the AAAI Workshop, AAAI Technical Report WS-99-05*. (pp. 59-64). *AAAI Press.*

Tiihonen, J., Soininen, T., Männistö, T., & Sulonen, R. (1996). State-of-the-practice in product configuration—a survey of 10 cases in the Finnish industry. In Tomiyama, T., Mäntylä, M., & Finger, S. editors, *Knowledge Intensive CAD*, volume 1, 95-114. Chapman & Hall.

Tiihonen, J., & Soininen, T. (1997). *Product configurators: – Information system support for configurable products.* Technical Report TKO-B137). Helsinki University of Technology, Laboratory of Information Processing Science. Also published in Richardson, T. (Ed.) 1997. *Using Information Technology During the Sales Visit*. Cambridge, UK: Hewson Group.