TRANSFORMING TELECOMMUNICATIONS SERVICE EXECUTION

Software is increasingly underpinning the services enterprises offer to customers and the operational processes which deliver and support those services. This is particularly true in the telco sector where innovative technologies such as Software as a Service, big data, cloud computing and Network Function Virtualisation (NFV) are further reducing the reliance of operators on specialised hardware for service execution and support.

SALWA ALZAMI, SID SHAKYA, IVAN BOYD

Telecoms and software

The increased speed with which services can be realised via software is a key contributor to the acceleration of telecommunications service innovation. Software also facilitates the easy adaptation and customisation of services, enabling operators to easily offer multiple variants of the same underlying service to different customer segments. However, whilst software is a key enabler of fast service innovation and flexibility, operators quickly discover that the maintenance of software is complex and very expensive.

A recent estimate by Accenture suggests 60% of global software spend (equivalent to ~$50Bn) is invested in software maintenance [1]. Thirty percent of this total software maintenance cost is associated with software comprehension. This is an astonishing figure given enterprises have either developed the software themselves or outsourced the software development. Why then do enterprises have to spend so much effort comprehending the software they commissioned?

Part of the answer lies in the inherent complexity of software itself. However, other major contributors are insufficient governance of the software development process and a lack of comprehensive documentation of the final software product. Time-to-market pressures habitually lead to compromises in the investment in documentation, which is often seen as something that can be completed later; unfortunately, that later occasion seldom occurs. Thus very often the only comprehensive documentation of software, which is essential to plan new versions and undertake maintenance, is the software source code itself. Source code created by one software engineer is often difficult for another software engineer to understand and follow.

It is important for the telco community to understand the latest developments in the software industry which are aimed at improving software governance and the automation of documentation creation. Telcos need to adopt these state-of-the-art tools and technologies to improve the quality and reliability of their services and to minimise the cost of software creation and maintenance, primarily by maximising reuse of software designs and the software that realises those designs. In this article, we present a review of one such technology, known as Software Product Line (SPL). SPL is increasing being adopted by some leading global enterprises to improve the speed of service development and service flexibility.

Software product line

SPL is an innovative software governance and reuse approach that introduces a systematic software planning strategy. This also helps enterprises address the challenges of software maintenance. The SPL methodology is focused on constructing similar software products from the same shared software assets, where the goal is to build highly customisable applications that can satisfy the needs of a particular business domain or customer segment.

SPL is an evolution of the traditional software development life cycle. It introduces a robust technique which enables the planning and creation of

reusable software units at the start of the software development. A key aspect of SPL, and one which distinguishes it from most other software development methodologies, is the comprehensive scope of reuse which SPL enables. SPL aims to reuse all the types of software assets and artefacts involved in the development lifecycle, such as user requirements, architectures and test plans, not just source code.

The concept of product lines dates back to 1968, when McIlroy [2] presented the idea of software mass customisation in order to make software components available to different machines and users. Later in 1972, Parnas [3] introduced the concept of developing program families, instead of developing products consecutively. He mentioned the importance of studying the common properties of the set of systems and then determining the unique properties of the individual family members.

Key concepts

SPL is defined by two development processes [4] – domain engineering and application engineering.

Domain engineering creates a reusable software platform whereas application engineering derives different versions of the product from this platform.

The reusable software platform consists of different software assets such as requirements models, architectural models, software components, test plans, and test designs. These software assets are defined as common or variable type depending upon whether they are compulsory or optional across different versions of the product.

The well-known multiple-views software architecture style in software engineering can be applied to SPL. Here, each different types of software asset can be structured in its own view, such as requirement view, architecture view, etc. Each of these views are normally owned by a stakeholder. The stakeholders can use the views to get a clear understanding of the current state of the system, and from their own specific prospective. The view can be defined succinctly and comprehensively using available modelling languages.

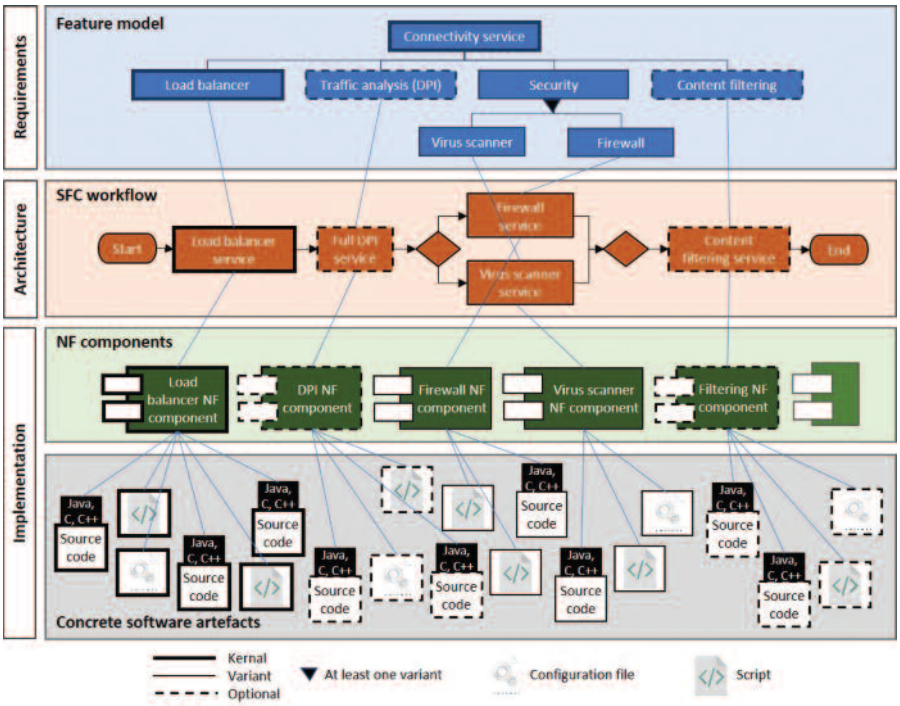

Figure 1: Multiple levels of SPL software assets for the NFV of a connectivity service.

As mentioned before, in domain engineering process, some of the software assets are considered as kernel (i.e.common assets to all products), while assets that vary across the products in the line are either identified as optional (nice to have) or variant (alternative of choices). A SPL variation point is placed in each asset to identify which variant has to be used for a specific software version.

A group of software assets defining a specific capability or functionality are grouped as a feature. These features are normally identified by a domain expert during a domain analysis process. Each new software product created by SPL is defined by a unique combination of features. The relationship between features are captured in a feature model which defines the product line.

As stated before, the feature model and the reusable software assets created during domain engineering process is fed to the application engineering process. Here, different features from the feature model are selected to determine which software assets will be included and configured in a new product. The output of this process is a collection of configured software assets, typically a new (version of) product or software, ready to be deployed.

Application

The companies that reported on their experiences of implementing SPL have listed a number of benefits, including significant reduction on time-to-market. For example, Boeing reported improved affordability, quality, and system timeliness by means of SPL. Market Maker Software AG reported that SPL helped it to reduce the maintenance costs by 60%. Cummins reported incredible reduction in its time-to-market, from one year to one week [5] [6].

SPL engineering is a growing software engineering sub-discipline, and many other organisations including Philips, Hewlett-Packard, Nokia, and Raytheon are using it to achieve extraordinary gains in productivity, time-to-market, and product quality.

How is the SPL methodology applied? – NFV application

To demonstrate the SPL methodology, we present an application from the telecommunications industry on building a customised virtualised network function platform.

NFV aims to address cost reduction and flexibility in network operations whilst enabling innovative network service delivery models. The network functions (NFs) are implemented as software running over a virtualised infrastructure and provisioned on a service-by-service basis. These NFs are chained together to provide network services which are deployed either in a data centre or via the cloud. In an NFV environment, a Service Function Chain (SFC) is used to describe the various NFs and the order in which they must be executed. NFV needs to adapt to ever-changing user and service requirements and the dynamic network context. By building a highly customisable NFV platform, less rework is required to deploy different versions of a service for different requirements and constraints.

NFV can benefit from SPL by adopting a systematic method for customising network services to accommodate diverse requirements from users and network providers. In particular, SPL can enable the auto-configuration of the SFC workflow to accommodate the different options available in different deployment. Additionally SPL facilitates the systematic organisation of the reusable NF components and the design of the SFC workflow such that the initial design encompasses both the kernel and optional connections and provides cost effective service chain configurations.

As shown in Figure 1, each SFC definition is considered as a product line that has:

1. Its own feature model (dotted boxes are optional features and bold are kernel features).
2. Its architecture model, consisting of one or more network services related to the features, defined in an SCF workflow.

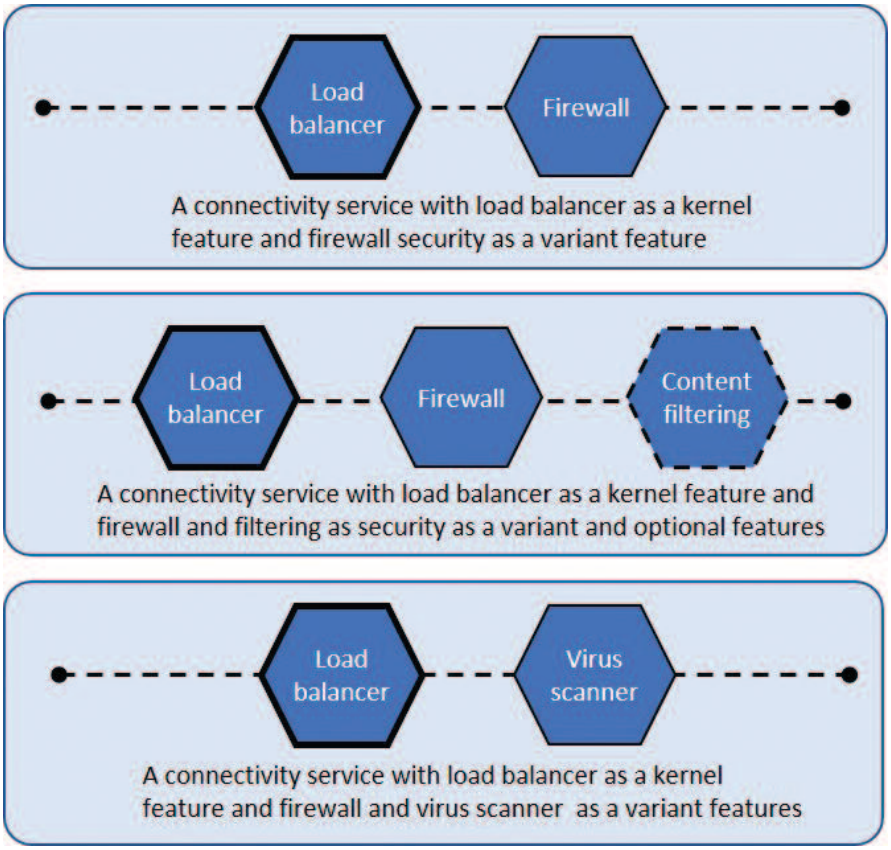

Figure 2: Multiple instances of SFC connectivity service.

3. Its NF components for each services.
4. The software artifacts (such as codes, scripts, configurations, etc) implementing the NF components.

Figure 2 shows three examples of SFC connectivity service instances that are derived from the original SPL SFC connectivity workflow (Figure 1). Based on the requirements, any one of them can be configured and deployed. As shown in Figure 2, load balancer as a kernel feature appears in all three SFC instances of the connectivity service, whereas the content filtering as an optional feature might appear in some instances. On the other hand, any connectivity service should have a least one security feature. Therefore, either the firewall and/or the virus scanner, which are variant features, should appear in any SFC connectivity instance.

The SPL approach to NFV implementation not only enables the proper planning of NFV platform software artifacts, but it also enables a robust customisation of SFC services that can adapt rapidly to situations, such as fluctuations in the network execution environment, service failures and change in end-user or NF vendors' requirements.

Some challenges

There are some challenges that needs to be addressed when adopting the SPL methodology. These include:

- The complexity of managing the variability information.
- Lack of suitable tools to fully support the SPL methodology.
- Legacy systems where knowledge of the architecture and software components is not available.
- The difficulty of implementing a new way of working when development teams may not want to move away from their current comfort zone.

In the next section, we have a tool that addresses some of these challenges is described.

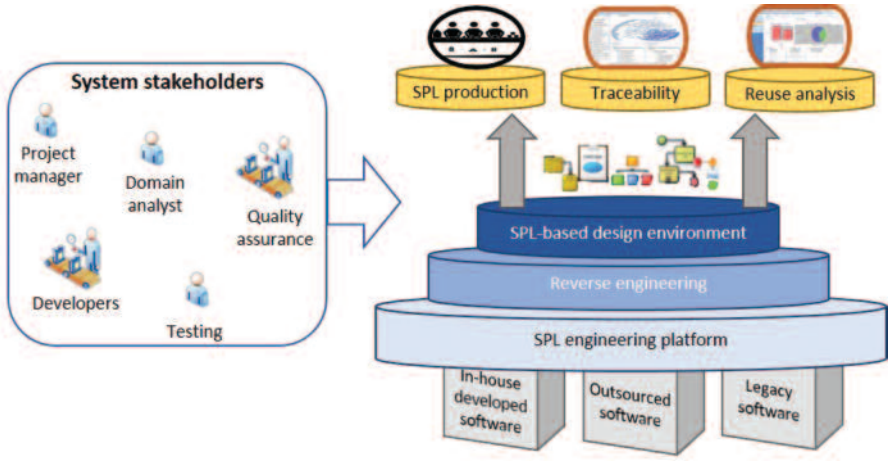

Figure 3: The services provided by EBTIC SPL tool.

EBTIC SPL approach

EBTIC is an ICT research and innovation centre established by Etisalat, BT, and Khalifa University and supported by the United Arab Emirates ICT fund. It is located at the Abu Dhabi Campus of Khalifa University with management and research staff from BT, Etisalat and Khalifa University. EBTIC aims to advance intelligent systems technologies for the Next Generation Networks and Next Generation Network-enabled ICT applications and services, in order to put in place the support infrastructure to facilitate, develop and enable the digital networked economy in the United Arab Emirates and beyond [www.ebtic.org].

EBTIC partners, BT and Etisalat, recognised the importance of software governance to their businesses. High quality software solutions, improved software comprehension and maximisation of reuse were recognised as key success factors in the delivery of innovative business services faster and with increased flexibility. Thus, BT and Etisalat commissioned research by EBTIC which lead to the development of the EBTIC-SPL tool described in this section and currently being trialled.

EBTIC's SPL tool is a novel agile solution that provides an in-house SPL-based software governance platform. It recognises the emergent needs of supporting product family development by telcos both for forward-development of new software systems and the maintenance and upgrading of legacy systems. The tool was developed specifically to overcome some of the challenges of adopting SPL listed at the end of previous section.

As shown in Figure 3, the four fundamental services that the SPL tool provides for system stakeholders are:

1. An SPL-based design environment that supports the SPL domain engineering process within an enterprise in order to develop the software assets of the product line. The environment enables the modelling, in multiple-views style, of common and variable software artefacts and maintains the traceability links between them. It accommodates different software architecture paradigms (object-oriented, component-based, and service-based architecture).
2. A reverse engineering capability that enables SPL implementation for the existing/legacy systems.
3. An SPL production that enables the creation of different products by auto-configuring the product line multiple-views design models (triggered through selection of the required feature set). The tool maintains consistency between product variants at the production process to ensure the integrity and validity of the final generated product. Model transformation techniques are incorporated in this process to implement changes in the concrete artefacts.
4. A traceability capability which captures traceability links between product line artefacts to support the various activities of the application production processes - these links also support the consistent evolution of the product line – and a reuse analysis capability incorporated at the backend to locate possible reusable and variable software assets.

What benefits does the tool deliver?

The tool addresses two key challenges facing telcos when developing software-based products, namely:

- Software design approaches that improve product quality and time-to-market. To accelerate concept-to-market requires a robust software design strategy that speeds software development and deployment. This replaces the ad-hoc techniques that focus only on code reuse.
- Automatic creation of comprehensive and easy to understand documentation. Without a full understanding of the developed software, a telco's goal of maximising reuse is exceedingly difficult. The tool delivers documentation which aids comprehension by employing multiple-views of the software system. These views communicate valuable information (requirements, architecture, process, and implementation) to all the stakeholders, not just software developers.

The EBTIC SPL tool is currently being trailed and has demonstrated the potential for improved software comprehension and governance. The BT pilot on FieldPlan solution, a component of BT's enterprise resource management platform called Field Optimisation Suite, has identified a number of challenges, such as those discussed earlier, for instance addressing legacy systems and adapting the SPL way of development. The trial was successfully able to overcome these challenges and

highlighted a number of benefits:

- Improved governance of outsourced software – as the structure of a software system can be captured in a platform-independent manner via reverse engineering of legacy code, the ability to bring outsourced work in-house or use a different supplier is significantly improved.
- Transformed way of productionising software assets – BT collaborates with a number of. The tool enables BT to have visibility of all the features in its software assets when engaging with the various stakeholders throughout the development cycle.
- End-to-end traceability – the capability of the tool to capture features and demonstrate end-to-end traceability improves the governance process and leads to better software assets reuse.

AUTHORS' CONCLUSIONS

As software increasingly underpins the services that telcos deliver to customers, the importance of understanding and applying the best of modern software engineering practices is critically important to service quality, speed of new service delivery and service flexibility. SPL is one of the most important modern software engineering approaches that telcos should be considering, given its success in delivering improved productivity in other software sectors. EBTIC has created a SPL tool which is being used to investigate the potential benefits of SPL to telco product development and maintenance. The initial results are encouraging and further trials are planned.

ABBREVIATIONS

NF	Network Function
NFV	Network Function Virtualisation
SFC	Service Function Chain
SPL	Software Product Line

ABOUT THE AUTHORS

Salwa Alzahmi is a research associate with EBTIC at Khalifa University, Abu Dhabi. She obtained her Master's degree from Khalifa University in 2014. She has over 9 years' experience in software engineering R&D. Her main research interests are in software product line and model-based engineering, with an emphasis on developing techniques and tools to improve software quality analysis. She has won a number of local and international prestigious awards. Salwa's goal is to turn software research findings into practical solutions.

Dr. Sid Shakya is a principle researcher at BT, currently working with EBTIC at Khalifa University, Abu Dhabi. He leads R&D activities in resource management and telecom services optimization. He has extensive experience of delivering intelligent software solutions to improve business processes. He is interested in practical applications of AI, particularly for business modelling and operational transformation. He has co-authored over 50 papers, filed 9 patents and won numerous awards for his work in applying AI techniques for industrial problems.

Dr Ivan Boyd is director of an independent telecommunications consultancy, having previously worked for BT for almost 30 years. During his BT career he managed global research teams and led a wide range of telecommunications research programmes from inception through to delivery. He now helps companies to optimise their research and innovation processes and secure Intellectual Property Rights. He also contributes directly to their technical research programmes. Ivan is a member of the ITP Editorial Board.

ITP INSIGHT CALL
Want to know more?
Join in the ITPInsight Call. Visit:
https://www.theitp.org/calendar/

REFERENCES

1. Huff, N., Glover, C., and Hillman, M. How Software Maintenance Fees are Siphoning Away Your IT Budget – and How to Stop It. 2014. Available at: https://spendtrends.accenture.com/wp-content/plugins/pdfjs-viewer-shortcode/web/viewer.php?file=https://spendtrends.accenture.com/wp-content/uploads/2015/05/Accenture-How-Software-Maintenance-Fees-are-Siphoning-IT-Budget-Procurement-BPO.pdf&download=true&print=true&openfile=false
2. McIlroy, D. Mass Produced Software Components. Proceedings of NATO Software Engineering Conference, Germany, pp. 79-85. 1968. Available at: http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF
3. Parnas, D. On the Criteria To Be Used in Decomposing Systems into Modules. Communications of the ACM, vol. 15, pp. 1053-1058, 1972. Available at: https://www.cs.umd.edu/class/spring2003/cmsc838p/Design/criteria.pdf
4. Pohl, K., Bockle, G., and van der Linden, F. Software Product Line Engineering. Springer, 2005
5. Case Studies. Software Engineering Institute, 2017. Available at: http://www.sei.cmu.edu/productlines/casestudies/
6. Krueger, C. and Clements, P. Systems and Software Product Line Engineering. BigLever Software, Texas, 2013. Available at: http://www.biglever.com/extras/PLE_SE_Encyclopedia_2013.pdf