Towards a Unified Framework for *Co-design* of Business System and Software System

V. Jamwal & H. Meda
TCS Innovation Labs, India

1. Introduction

Modern businesses depend upon Software Systems for competitive advantage. Success of Software System, in turn, depends upon how well it is able to align with the present and growing needs of business [1]. Traditionally the design of the two systems, viz. Business System and Software System, is done by a different set of persons. Business Systems are mainly designed by business persons – CEO, COO, strategy team, product owners, process owners, marketing etc. The Software System design is mainly handled by IT persons – CTO, Software architects, ADM (Application Development and Maintenance) teams, etc. Because of the environment in which they operate, often these two sets of persons support different worldviews. We argue that for creating effective Business and Software System alignment, these two systems should be *co-designed*. Further we believe that if *co-design* can follow a unified design paradigm, these worldviews can be more effectively bridged. Does such a unifying paradigm exist? We think it does. This position paper explores such a unifying paradigm and tries to lay a foundation for *co-design* of Business System and Software System.

2. Four Causal Factors

Greek philosopher and polymath, Aristotle (384 BC – 322 BC) opined that reason for every '*thing'* coming about can be attributed to four different types of simultaneously active causal factors: (i) *Material Cause* - that out of which a thing comes to be and which persists, (ii) *Formal Cause*: the form into which something is made, (iii) *Efficient Cause*: that by which something is produced, and (iv) *Final Cause:* that for the sake of which a thing is done. Refer [2] for a more detailed discussion on this topic.

We build on this observation. We state that any System (whether it produces product or service) can also be treated as a '*thing*' and we can associate the four causal factors to it. How do these four causal factors unfold? We take the example of a *Car Manufacturing System* to illustrate this: (Refer figure 1)

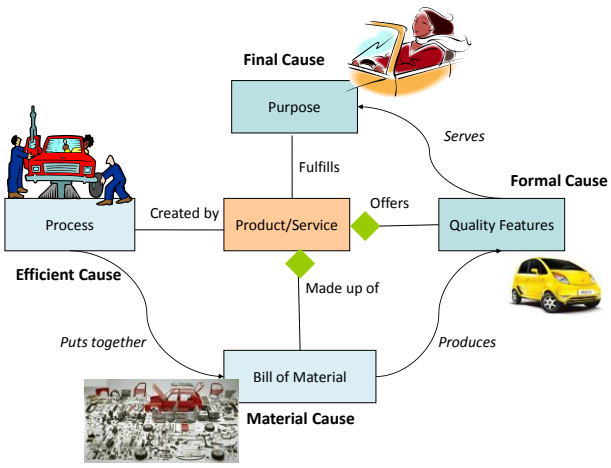

Figure 1: Four Causal Factors

(i) A car is composed of components - chasse, engine, transmission system, wheels etc. (bill of material). This can be said to constitute the *material cause*.

(ii) The assembly of these components yields quality features for the car – mileage, safety, suspension, cost, etc. This can be said to constitute the *formal cause*.

(iii) The components of the car are manufactured and then assembled together in a car-factory following a definite process (e.g. chasse assembly, body assembly, painting, etc.) and utilizing various tools. This can be said to constitute the *efficient cause*.

(iv) Finally, the car is shipped to the customer and serves her needs of transport and leisure. This can be said to constitute the *final cause*.

3. Implication of Causal Factors on System Design

Generalizing from the above example, we state that a *System (product/service)*: is made up of *product/service* components; is assembled/delivered through a *process*; offers a set of *quality features*; serves the needs of user (*customer value*).

We further state that each of these causes need not be treated as isolated factors. They can be further linked with each other. These linkages provide the *design-basis* for the entire System. For example, the design implications for the car-manufacturing-system are: The features of the car should be so designed that they serve the needs of customer. The component configuration of the car should be so designed that the requisite quality features for the car are produced. The manufacturing process should be so designed that each component is produced with requisite attributes and is assembled correctly.

We have developed a framework, *f*-QDF (*fine-granular* Quality Design Framework)[3], which builds on these core principles. Our initial experience has shown that this model can be applied to all *purposive systems* [4]. Since Business System and the Software System are both purposive in nature, they can be designed following a unified design paradigm.

4. Workshop Participation and Aims

During the workshop we aim to illustrate with the help of a Case-study how *f*-QDF is applied to Business System design. We also wish to highlight *how* the framework creates the following distinct advantages in the design of Software Systems:

(i) It helps to integrate various aspects of Software System Design – User Experience, Feature Design, Component Design, and Software process. Refer figure 2.

(ii) It provides a better understanding of the User Space, leading to sharper requirements for Software System design.

(iii) It facilitates *co-design* of Business System and Software Systems. Refer figure 3.

The *f*-QDF Framework arose out of authors' work with Business System transformation projects and was conceptualized in early 2010. It has since been applied towards design and transformation of Business service systems, including document processing service of a major consulting company [3]. At present it is being applied on the Trade Financial Systems of a multinational financial institution. Our first goals are to refine this framework for Business System design – so that it serves as a robust model for business transformation and software system requirements elicitation. We are also planning to apply it on the Software System design, particularly in the context of Model Driven Engineering. The Framework is in the early stages of industrial validation and adoption. Through the discussions with other participants at xDD SPLASH Workshop, the authors hope to (i) add to their understanding of software design challenges faced by practitioners (ii) receive feedback from the design community in further shaping the Framework.

References

[1] Michael Hammer, James Champy, "Reengineering the Corporation: A Manifesto for Business Revolution", HarperBusiness; Rev Upd edition (December 23, 2003)

[2] Falcon, Andrea, "Aristotle on Causality", The Stanford Encyclopedia of Philosophy (Fall 2011 Edition), Edward N. Zalta (ed.),URL=`http://plato.stanford.edu/archives/fall2011/entries/aristotle-causality/`

[3] V. Jamwal and H.S. Meda, "fQDF: A Design Framework for fine - granular Quality Control of Business Process Outcomes", Business Process Design Workshop, BPM 2011,Clermont-Ferrand, France, August 29, 2011.

[4] Jamshid Gharajedaghi, "Systems Thinking, Managing Chaos and Complexity: A Platform for Designing Business Architecture", Morgan Kaufmann; 3 edition, July 13, 2011.

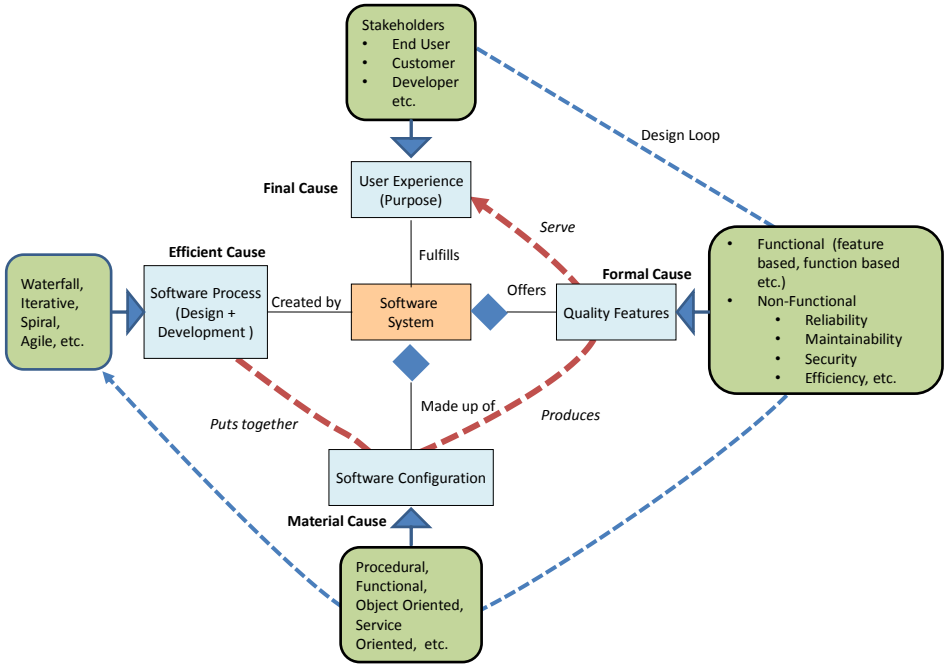

Figure 2: Software System Design Spaces

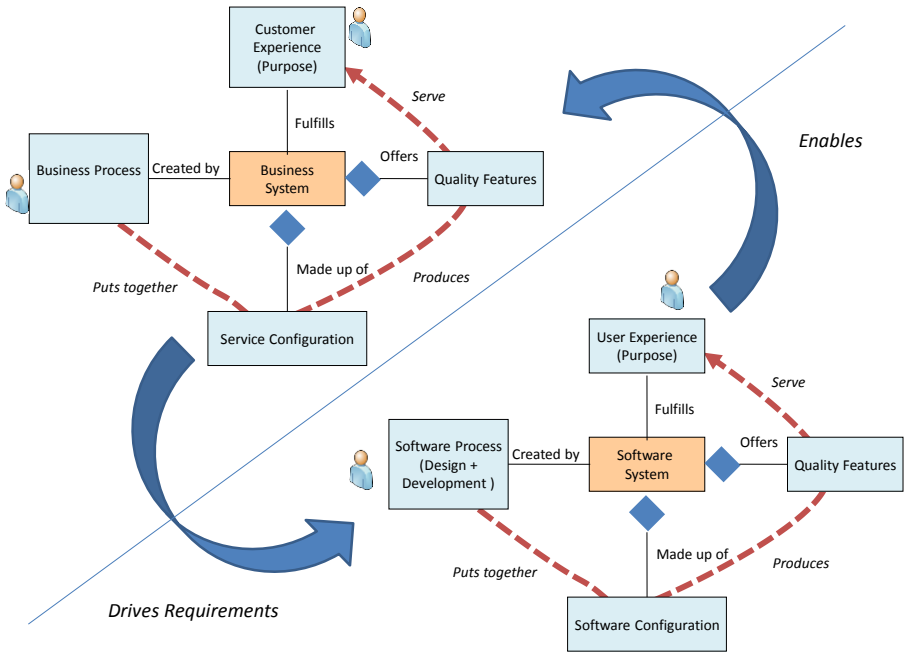

Figure 3: Business System - Software System Co-design and Alignment